# An Edge Service Architecture for Multimedia over Wireless Networks

Rajesh Mahindra Ravi Kokku Honghai Zhang Sampath Rangarajan

Mobile Communications and Networking Research
NEC Laboratories America, Princeton.

## Abstract

*In this paper, we describe the design and instantiation of a multimedia edge service architecture (*MESA*) for enhanced video transmission over wireless networks.* MESA *encompasses simple and novel rate-matching and shaping mechanisms as building blocks that ensure that the video rate matches the instantaneous available wireless channel capacity at fine timescales. Designing such a rate matching scheme as a service at the wired-wireless edge (transition point) enables three distinct advantages: it (1) achieves high accuracy leading to better video quality, (2) facilitates immediate deployment, and (3) enables a generic design applicable to several wireless technologies, with the augmentation of a few technology-specific mechanisms. We implement a* MESA *prototype and evaluate it on both WiMAX and WiFi testbeds using static and mobile client scenarios. Our results show that* MESA *selectively drops up to 5% low priority frames to improve the PSNR of video sessions significantly—by 7 dB on an average. By enabling a WiMAX-WiFi handover, we demonstrate that* MESA *adds negligible overhead to intra- and inter-technology handovers.*

## 1. INTRODUCTION

Recent years have witnessed a rapid proliferation of sophisticated mobile devices that can capture-and-stream [13] and receive-and-view [9] high-quality video, even in real-time. These devices are often connected over one or more wireless networks on the last-mile. This is leading to an increasing interest among mobile network operators (MNOs), mobile virtual network operators (MVNOs) and content providers (CPs) in enhancing streaming video services and video telephony on wireless networks. Two emerging trends further fuel this interest among the supplementary equipment providers [7, 10, 18] of MNOs and MVNOs: (1) the improvements in data transmission rates of wireless networks (e.g. 1 Gbps target rate for nomadic users and 100 Mbps for mobile users in 4G systems [11]), and (2) the advancements in video compression techniques and standards (e.g. H.264/MPEG-4-AVC and its scalable video coding extensions [37]).

Despite the advancements, however, the increasing number of users demanding services that involve video [13, 16], and the scarcity of the frequency spectrum together preclude *over provisioning* of wireless resources

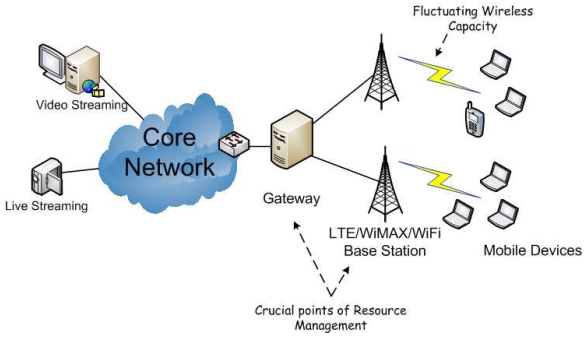



**Figure 1: Typical wireless network deployment.**

per-user through bandwidth reservations for maximizing user-perceived quality. Since wireless channel capacity is often significantly lower than the capacity on the wired side, consequently, the wired-to-wireless transition point becomes a crucial place for efficient resource management (See Figure 1). In this context, a key resource management consideration at the transition point for effective streaming to mobile clients is to dynamically *match the rate* of video streams (and hence the user perceived quality) to the available capacity on the wireless link.

Effective rate matching, although conceptually simple, requires addressing several challenges. Firstly, video rate requirements fluctuate significantly with time due to the difference in content even in adjacent video frames. Further, the differences in codecs, the inter-dependence of frames, the packetization of frames for transmission over the Internet, the aggregation of different frames into the same packet, and the effect of individual packet loss on an entire frame, may require that rate matching be done at fine (e.g. per-packet) granularity. Secondly, wireless channels are affected by fading, user mobility, variation in number of users, traffic generated by the users, and the base station scheduling policies used, thereby leading to significant variation in per-user channel capacity at fine-timescales. These characteristics require that rate matching be *agile*, while causing minimum extra overhead on the existing infrastructure to be easily deployable.

Video delivery over wired and wireless networks have received significant attention in the past; solutions proposed can be categorized into end-to-end [24, 25, 27], dynamic in-network transcoding [38] and base station scheduling [19, 33, 32, 41] approaches. However, while

end-to-end approaches are slow in reacting to fine timescale fluctuations in video rate and wireless channel capacity, in-network transcoding incurs significant overhead thereby limiting its applicability to only coarse timescale adaptation, and base station scheduling approaches have been either too close to a specific wireless or video coding technology, or have overlooked several system design issues.

In this paper, we design and implement an in-network solution called MESA that encompasses simple and light-weight rate matching and traffic shaping mechanisms to match the video rate to the instantaneous wireless channel capacity in the downlink direction. MESA has three salient features: (1) being close to the wireless channel, it can make intelligent choice of frames to transmit based on the available channel capacity at fine timescales, resulting in minimum distortion in video quality, (2) it can be deployed as an *edge service* without much dependence on the specific wireless channel access technology such as WiMAX, LTE or WiFi, and (3) minimum dependence on the client and server modifications facilitates quicker deployment. We also extend MESA to enable handover of an active video session when a user migrates from one base station to another (e.g. from WiMAX to WiFi, Macro- to Femto-base station, etc.).

We evaluate MESA on (i) a mobile WiMAX (802.16e) network testbed containing the NEC WiMAX ASN gateway (that runs MESA), the NEC WiMAX base station [6], and a few Beceem-chipset-based clients [1], and (ii) a WiFi network testbed with a controller (that runs MESA), an AP and a few clients. Extensive evaluation shows significant improvements in video quality with MESA under several conditions on both testbeds. For example, in the WiMAX setup with one H.264 AVC video, while no rate matching led to a total of about 5% late frames, 12% lost frames, and 25% corrupt frames, carefully dropping 5% of B frames led to <3% of late frames and no lost and corrupt frames. This led to a PSNR improvement of 7dB and a marked difference in video quality visually. Also, our handover experiments from a WiMAX- to a WiFi base station demonstrate that MESA itself adds negligible overhead to the handover process, and provides seamless continuity of video transmission.

In summary, we make three main contributions:

1. We propose a simple and light-weight edge-service solution for enhanced delivery of unicast live streaming content, whose *agility* to fine timescale fluctuations in video rate and wireless channel capacity improves user-perceived quality significantly.

2. As a part of technology-specific mechanisms, we design a novel approach for WiFi networks to identify accurately if the channel is at saturation. We believe that this mechanism is useful beyond MESA.

3. This is the first work that presents implementation experiences and evaluation results on a WiMAX testbed, to the best of our knowledge. While several features of WiMAX are unique, our learning suggests that solutions that are designed to be independent of the channel access method are equally applicable on WiMAX and WiFi networks.

The rest of the paper is organized as follows. Section 2 formulates the problem by discussing the design considerations in detail and the drawbacks of related work. Section 3 presents our design. Section 4 discusses extensions for supporting video continuity during handovers. Section 5 describes our prototype, and Section 6 discusses evaluation on WiMAX and WiFi testbeds. Section 7 concludes with a discussion.

## 2. DESIGN CONSIDERATIONS

Much research has gone into the enhancement of video delivery over the Internet (See [21], [29] and references thereof). To help place existing solutions in the right context, we first understand the domain characteristics of the emerging video and wireless network technologies, and then discuss the requirements of an effective solution for enhanced video streaming.

### 2.1 Domain Characteristics

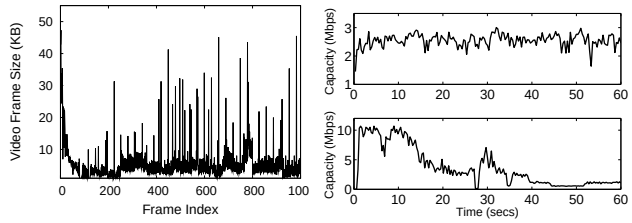
For completeness sake, we start with a discussion of several characteristics that influence the quality of video streaming as perceived by the users; these observations are not new and have been repeatedly mentioned in past works (e.g. [21, 29]). We consider three videos of different characteristics, as shown in Table 1. All videos are coded under the H.264/MPEG-4 AVC—a popular standard that allows compression of videos by inter-frame encoding. As a result, some frames are much larger than others based on how they are encoded. Specifically with H.264 AVC, the frames are divided into a sequence of GOPs (group of frames), where each GOP is a sequence of three types of frames I, P and B, which are transmitted in the order [I P B B P B B... ]. The I frame is independent and contains complete information, each P frame is coded relative to the previous I frame or P frame, and each B frame is coded relative to the previous I or P frame.

**Fluctuating video rates:** Figure 2(a) shows the number of bytes per frame in the Sports video. The graph shows that the number of bytes that need to be transmitted varies widely with time. We can see from Table 1 that I frames can be as high as 6 times larger than B frames (for Sports) on an average, and the peak to average ratio of frames within one type can be as high as 9.4. These observations demonstrate that the bandwidth requirement of a user requesting a video stream fluctuates significantly over time.

**Fluctuating wireless channel capacity:** In general, wireless channel capacity varies with fluctuations of channel conditions due to fading and shadowing, bit-rate adaptation, user mobility, changing number of users, channel contention, etc. As an illustration,

| Video | Avg Frame Size (KB) | | | Peak to Avg Ratio | | | Avg. rate | Description |
|---|---|---|---|---|---|---|---|---|
| | I | P | B | I | P | B | (Mbps) | |
| User | 15.7 | 6.65 | 3.45 | 2 | 1.98 | 2.45 | 1.16 | A user-captured video with slow changes. Representative of typical video telephony and conferencing |
| Sports | 24.1 | 7.8 | 3.7 | 1.91 | 4.11 | 8.6 | 1.36 | A car racing video clip with slow-to-fast changes. Typical of live streaming of events. |
| Trailer | 16.5 | 8.4 | 4 | 4.9 | 3.4 | 9.4 | 1.42 | Movie trailer with rapid changes. To emulate fast changing events. |

**Table 1: Characteristics of different types of videos used in the paper.**




(a) Video Rate (b) Channel Capacity

**Figure 2: (a) Variations in size of frames for the sports video. (b) Fluctuation with WiMAX for static (top) and mobile (bottom) clients.**

Figure 2(b) shows measurements in a WiMAX network with the base station and clients transmitting over the air. A single client transmits in the 2.585-2.595 GHz channel, for which NEC Labs has obtained an experimental license from FCC. We reduce the transmit power of the WiMAX base station to match the range of WiFi. The top graph shows the fluctuation in downlink throughput[1] achieved by the client when the client is placed statically 20 meters away, while the bottom graph shows the throughput when the client starts close to the base station and is moved 40 meters away at walking speed. The graphs show both short-term and long-term variations in throughput. We make similar observation on WiFi.

## 2.2 Implications and Design Requirements

**Functional Separation and Adaptation Granularity:** While long-term variations in available bandwidth for a user are better handled by transcoding (at source or intermediate node) and/or renegotiating bandwidth requirements [22, 31], short-term fluctuations may be handled by prioritized video frame dropping [32, 33, 41], preferably as close to the wireless channel as possible. This is because the fluctuations may last for much shorter durations than the time taken by the video source to receive feedback and react accordingly. To facilitate effective frame drops when necessary, ideally, the video source identifies and marks each packet with the right priority level (based on the video frame it carries). The intermediate node only uses the marks to make dropping decisions. Such "separation of concerns" also ensures greater security of the application-level content. For instance, the video may be encrypted end-to-end, as long as the packets are marked with the right priority levels. Further, this design decision ensures minimum modifications to end-hosts to handle short-term fluctuations that are more specific to the wireless domain.

**Flexibility and Reusability:** Solutions for video enhancement are often independent of the specific wireless technology, and should hence be designed with flexibility and reusability as goals. This broadens the applicability of a solution for several reasons. Firstly, Network equipment manufacturers often build devices for multiple technologies [8, 12]; reusing such solutions reduces overall development time for enhanced services. Secondly, mobile network operators increasingly focus on heterogeneous last-mile technologies to provide high bandwidth and coverage to multi-mode mobile devices (e.g. see T-Mobile's hotspot@home [14], and the rationale behind Femtocells [8, 12]). Finally, future MVNOs that provide such enhanced video services may change MNOs over time or run over multiple MNOs (even across heterogeneous wireless technologies such as LTE, WiMAX and WiFi) for providing continuous coverage to their customers. Consequently, a solution should separate technology-independent and technology-specific components to enable reuse over multiple wireless technologies. Further, this separation also simplifies the handover process when a user migrates from one base station to another.

## 2.3 Related Work

The effects of fluctuations in available capacity and video rate were handled in several ways in the past. Related work can be categorized into end-to-end approaches and in-network approaches. While there is a plethora of works, we discuss a sample set of efforts (for brevity) to place MESA in the right context.

**Sender-side adaptation:** Some end-to-end approaches involve receiving feedback at the source on the link characteristics or perceived video quality at the receiver, and adapting video transmission at the source [24, 25, 27, 42]. Kim et al. [27] use a probe-based channel adaptive video streaming over 3G using RTCP feedback. Juan et al. [24] discuss scalable video streaming over multiple connections in WiMAX. They propose

[1]Throughout the paper, we use throughput and capacity inter-changeably to mean the capacity that is achievable by an individual user on the wireless link under different conditions (such as modulation rate, mobility, etc.).

source rate adaptation (to adapt the video frames generated by the source) based on the feedback from the base station. The authors in [25] consider a non-fading environment (i.e., the channel gain or user throughput is fixed), and also consider source rate adaptation. An extreme case of these solutions is bandwidth smoothing algorithms that precode videos to smooth the bandwidth requirements a priori as much as possible [36, 23]. Feng et al. provide a detailed study of six classes of such bandwidth smoothing algorithms [21]. Bandwidth smoothing makes the video less bursty at the cost of quality. Overall, such sender-side adaptation techniques are not responsive enough to short-timescale fluctuations in the wireless channel.

**Network Adaptation:** Alternately, a few end-to-end approaches adapt bandwidth reservations to adapt to video rates. For instance, Song et. al. [39] and Grossglauser et. al. [22] propose intermittent bandwidth renegotiation from the network to adapt to changing video requirements. These techniques may be applied at coarse timescales for better admission control and statistical multiplexing across flows, but can not handle the inverse problem of adapting video rates to the throughput variations on the wireless channel.

**In-network transcoding:** These approaches involve modifying the video stream at an intermediate node in order to adapt to the link capacity variations. In [38], authors discuss the advantages of in-network transcoding over source-based video adaptation. However, transcoding involves significant amount of computation [38], signaling to inform the receiver of the change in coding, and also requires adaptation over a group of frames, which may not be responsive to fine-timescale fluctuations of wireless channel.

**Receiver-side Adaptation:** Some end-to-end approaches include adapting the playout rate to minimize effect of fluctuations on quality at the cost of increasing the playout time [17, 20]. While stretching the video in time may be tolerable for stored video playout, it is not desirable for interactive video (such as video telephony) and live streaming.

**Basestation Scheduling:** An alternative to in-network transcoding is to drop low priority frames to make enough room for high priority frames. This approach is often implemented in past solutions on the base stations in conjunction with scheduling [32, 33, 41]. Liebl et al. [32] proposed one such scheme for multi-user streaming. Their frame dropping strategy is based on the fullness of a buffer of maximum fixed capacity of N frames. IFD [19] is a similar solution, and is specifically tailored to H.264 AVC, and does not extend in a straightforward way to other coding schemes. Such solutions based on fixed buffers are ineffective with fluctuating network capacity; varying sizes and delays of frames could cause different deadlines on different frames. Hence even if the buffer is not full, there could be frames whose deadline is violated. In [33], the authors implement a deadline-based adaptive retry mechanism for a video flow in 802.11 WLAN context. They define different deadlines for I-,P- and B-frames based on priority and the MAC layer retries the packets until their deadline. But their mechanism does not take into account instantaneous conditions, and hence would be ineffective when the instantaneous video rate is above the available wireless capacity; e.g., in some cases, immediately dropping low priority frames is a better idea than retrying fewer times.

# 3. MESA DESIGN

In this section, we describe the design and implementation of an in-network rate matching architecture, called MESA, with three salient features: (1) being close to the wireless channel makes MESA agile; it can make intelligent choice of frames to transmit based on the available channel capacity at fine timescales, resulting in minimum distortion of video quality. (2) it is designed as an edge service without much dependence on the wireless channel access technology that fosters reusability, and (3) minimum dependence on end-host modifications facilitate immediate deployment.

Figure 3 shows an overview of MESA, whose approach involves three key components. First, it maintains a per-flow *buffer of frames* ordered by the earliest-deadline first policy. Second, a *rate matching module* determines for every new frame arrival (i.e. the arrival of *all* packets of the new frame) into the buffer, the effective service rate needed to meet the deadlines of all frames. If the service rate is higher than the wireless channel capacity for the flow, low priority frames are dropped to make room for high priority frames. Finally, an *adaptive shaper* ensures that the video buffer gets serviced only at the rate allowed by the instantaneous wireless channel capacity, in order to avoid buffer overflow at the base station.

In the rest of this section, we first identify the minimal application level extensions required, and then present the design of each of the components of MESA.

## 3.1 Application Extensions

We assume that the video source provides the following minimal extensions. Since video sources also care enough that best quality video reaches the users in the presence of fluctuating network conditions, we believe these extensions are readily doable.

**1. Packet marking:** We assume that the video source marks packets with appropriate priorities under different modalities of video coding (such as H.264 AVC or SVC). The priorities are marked such that 1 represents highest priority, and the priority decreases with increasing value of the marks, thereby ensuring scalability to arbitrary number of levels. For example, for H.264/MPEG-4 AVC with frame sequence [I P B B P B B ...], packets may be marked as 1 for I frames, 2 for P frames, and 3 and 4 respectively for the first

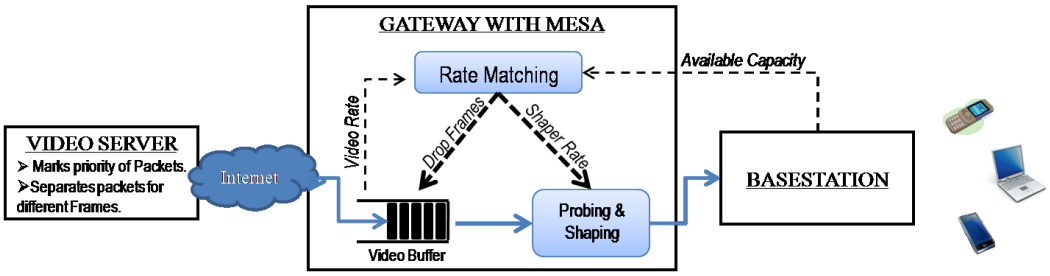



**Figure 3:** MESA **Overview.**

and second B frames after a P. Note that specifying different priorities for B frames also helps in identifying frame boundaries. Audio frames take the same priority level as the I frames so that audio distortion is minimized. For scalable video coding (SVC), the base layer is marked as 1, and the subsequent lower priority layers are marked with increasing values. These marks may either be included in the unused bits of the TOS field of the IP header of each packet, or included in the IP Options. We take the former approach in this work.

**2. Separating composite packets vs. Priority Inheritance:** While frames may often be split across several IP packets, some packets may contain multiple frames. For instance, streaming the Trailer video with VLC [17] leads to 45% of packets containing more than one type of frames. The same is true with video sources that combine video frames and voice frames into same packets; note that missing voice frames have more detrimental effect than missing video frames of the *same size*. Identifying frame boundaries is crucial for ensuring that we do not drop part of a high priority frame that is aggregated with a previous low priority frame. One approach is to ensure that packets contain only one frame type. Many SIP-based applications (e.g. Linphone [3]) already do the separation. This, however, comes at the cost of increasing the number of packets transmitted from the source. The second approach is to let each packet inherit the priority of the highest priority frame it carries. This approach reduces the granularity of frame dropping, especially at low rates. In this paper, we assume that the source takes the former approach to allow for finer granularity packet dropping.

## 3.2 Pre-Buffering

For each *flow*[2] from the wired side to a mobile client, we maintain an intermediate buffer that stores packets arriving from the wired interface before transmitting. Additionally, we delay the transmission of the first packet by $\mathcal{D}$ amount of time. Since frame display deadlines are typically relative to their previous frames, the delay has a positive effect of increasing the deadlines of each frame by $\mathcal{D}$. Pre-buffering enables MESA to initialize its parameters for effective rate matching. While this approach has minimal impact when the capacity is sufficient or higher than the incoming video rate, it leads to increased flexibility of *selecting* which packets to transmit/drop on the wireless link when the capacity due to fluctuations is lower than required. Note that for this reason, this approach is not equivalent to just increasing the *receiver's buffer* by the same amount $\mathcal{D}$, which does not allow choosing the packets to drop.

While higher values of $\mathcal{D}$ lead to smoothing fluctuations better and allow for more effective rate matching, they also lead to increased initial wait time in the case of live video streaming, and continued delayed transmission in the case of video conferencing, which could cause annoyance to the users. Hence, MESA restricts $\mathcal{D}$ to small values (200 ms in the prototype) to keep the impact low.

[2]We define *flow* as a UDP data connection carrying video data between the source and the user. The flow is identified using the five-tuple in packets including the source and destination addresses and ports, and the protocol.

**Algorithm 1** Rate Matching Algorithm

```
1: On arrival of a new frame OR
   On the departure of a critical frame OR
   On capacity reduction
2: if at_saturation () then
3:    V_r ← get_video_rate()
4:    A_c ← get_available_capacity()
5:    if V_r > A_c then
6:       Drop Frames
7:    end if
8: end if
```

## 3.3 Rate Matching

The goal of rate matching is to compare the effective incoming video rate $V_r$ (after incorporating the frame deadlines) and the available channel capacity $A_c$ on the wireless link, and drop low priority frames if the channel capacity is lower. Algorithm 1 depicts the overall behavior of rate matching. The algorithm checks for rate matching under three conditions: (1) when a new frame arrives into the buffer, (2) when a *critical frame* (which we will define soon) departs, and (3) when the estimated capacity on the link decreases. Rate matching is performed only when the channel is *at saturation*.

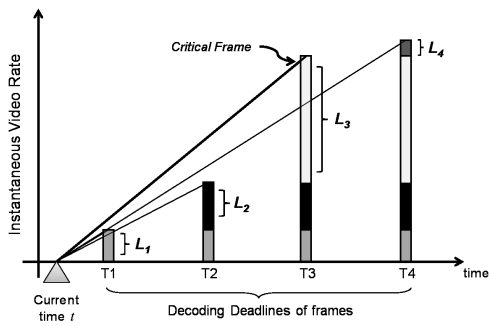



**Figure 4: Rate matching process.**

### 3.3.1 $V_r$ estimation

We define the incoming video rate $V_r$ as the rate required to meet the deadline of all frames in the intermediate buffer. In particular, for the $j$th frame in the buffer, let $T_j$ be the deadline and $L_j$ be the size of the frame. If $t$ is the current time, then $V_r$ for the user is

$$V_r = \max_j \frac{\sum_{k=1}^{j} L_k}{(T_j - t)} \quad (1)$$

The idea of this definition is to account for the fact that *any* frame in the buffer, and not necessarily the last frame, can determine the maximum required channel capacity to ensure that all frames meet their deadlines. Pictorially, in Figure 4, the slope of the lines indicate the rate requirement for a particular frame and its predecessors to meet their deadlines. Clearly, the third frame in Figure 4 determines the maximum rate required at the current time instant $t$; we call such a frame the *critical frame* till it is transmitted or dropped. Dropping low-priority frames before the critical frame in the buffer reduces the slope for the critical frame, which is the idea of rate matching.

**$T_j$ estimation:** Determining $V_r$ correctly involves determining the individual frame deadline $T_j$, which depends on the receiver buffer, the end-to-end network delay, and the frame display rate. One approach would be to make the video source include the value of $T_j$ relative to the first frame in each packet using IP Options. Another approach that we take in MESA is to assume that (1) the wired network does not add too much *variance* in delay when compared to the receiver buffer size, and (2) the source already performs flow control based on the actual deadlines and long-term average link capacity, and hence the deadlines can be approximated by the arrival times ($a_j$) of first packets of each of the frames, which are identified by change in frame types. Hence, we set the deadline for each frame as

$$T_j \quad = \quad a_j + \mathcal{D} + \mathcal{R} \quad (2)$$

where $\mathcal{D}$ is the extra delay we added due to prebuffering, and $\mathcal{R}$ is the receiver buffer size. The receiver buffer size can be set by making a conservative estimate based on commonly used values. This definition makes each frame independent of previous frame transmissions, thereby avoiding any accumulating effects due to channel variations and frame losses.

**Algorithm 2** *at_saturation()* for WiMAX

```
1: AR_DL ← snmp_query(Available_Resource_DL,BSID)
2: return (AR_DL < 5%)
```

**Algorithm 3** *get_available_capacity()* for WiMAX and WiFi

```
1: bytes ← query(MAC_Throughput_MS,FLOWID)
2: rate ← (bytes − last_bytes)/τ
3: last_bytes ← bytes
4: return rate
```

### 3.3.2 $A_c$ estimation and at_saturation()

Due to fluctuating channel conditions, it is not possible to determine how much capacity is available on a link at any instant of time. Consequently, it is often common to revert to indirect approaches such as using the throughput achieved in the immediate past to estimate the minimum future capacity available to the user. Such capacity estimation can be done in several ways depending on the medium access technology. To keep MESA's design general, we abstractly define two functions—*at_saturation()* and *get_available_capacity()*—that can be implemented in different ways in different environments. The former detects if the channel is completely utilized by the various flows or if there is spare capacity at the base station, while the latter returns the available capacity *per flow* at that instant. We now describe instantiations of the functions for WiMAX and WiFi networks.

**WiMAX:** Due to the slotted structure of WiMAX MAC, capacity estimation can be done with the radio resource parameters commonly maintained by base stations. For example, the NEC base station maintains several parameters that can be accessed via *http* and *snmp*. We use two parameters—*Available_Resource_DL* that gives us an estimate of the total downlink spare capacity in the base station as a percentage of average number of empty slots in the last few WiMAX OFDMA frames, and *MAC_Throughput_MS* that provides an estimate of the total bytes successfully transmitted to a mobile station for a specific flow. We show in Algorithms 2 and 3 the instantiations of *at_saturation* and *get_available_capacity* functions for WiMAX. The function *at_saturation* returns true if the empty slots are less than a threshold (5% in our prototype). The function *get_available_capacity* is called every $\tau$ units of time, and returns the rate at which packets of this flow were sent in the last $\tau$ units of time.

**WiFi:** With current WiFi hardware, an individual base station cannot determine if the channel is at saturation since the channel is shared, and access is con-

**Algorithm 4** *at_saturation()* for WiFi

1: **return** $(1 - \frac{get\_available\_capacity()}{get\_ett\_capacity()} < 5\%)$

**Algorithm 5** *get_ett_capacity()* for WiFi

1: ett ← EWMA of ETT per packet
2: ett_capacity ← pkt_size/ett
3: **return** ett_capacity

tention based. Hence, to identify the saturation condition, we use the following indirect method: we estimate the capacity using two different approaches and compare them (see Algorithm 4) with the idea that both the approaches return similar estimates only *in saturation*. Specifically, the available capacity with *get_ett_capacity()* is accurate only when the system is "actually" in saturation; otherwise the capacity is an over-estimate. The *at_saturation* function returns true if both rate estimations are within 5%. The function *get_available_capacity* is same as in Algorithm 3. We implement the query function in the MadWiFi driver on the base station[3] that returns the per-flow bytes successfully transmitted in the last $\tau$ units of time.

We realize *get_ett_capacity()* in the MadWiFi driver on the base station using the estimated transmission time (ETT) per packet as shown in Algorithm 5. To determine ETT of packets, we borrow the technique used by Symphony [35]. The key idea is to estimate ETT as the difference between when a packet is queued by the MadWiFi driver at the WiFi interface card, and when the card returns *tx_complete()* for the packet. Note that since the card can have more than one packet outstanding in its buffer, the ETT estimation becomes challenging, which has been addressed by Symphony effectively. One change we do to Symphony's approach is that Symphony only considers packets that succeed in the first transmission (without retries) to detect channel access asymmetry, whereas we consider all packets including those retransmitted since our goal is to estimate available capacity. We normalize the ETT of packets of different sizes to the same packet size of 1300 bytes. This approach estimates capacity in saturation correctly with both intra-node (i.e. inter-flow on the same node) and inter-node contention.

To demonstrate the efficacy of *get_ett_capacity()*, we show the result of an experiment (on channel 6 in IEEE 802.11g), in which the base station estimates the capacity for downlink flows to two clients. The base station transmits at 6 Mbps fixed rate; hence the maximum aggregate capacity to the two clients should be about 5 Mbps. Using iperf, the offered load of the first flow is incrementally changed and that of the second flow is kept constant at 2 Mbps. The graph in Figure 5 shows that the capacity estimate for both the flows is high (i.e., an over-estimate) when the link is

[3]Throughout the paper, base station in the WiFi context refers to the access point.

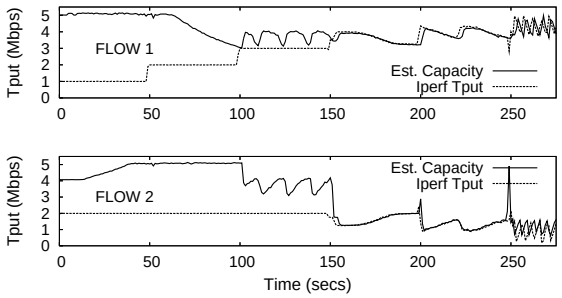
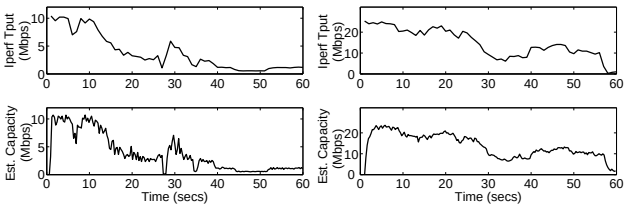



**Figure 5: ETT-based capacity estimation.**




(a) WiMAX (b) WiFi

**Figure 6: Available Capacity Estimation.**

not saturated, and becomes more and more accurate as the link approaches saturation. The estimate becomes precise after around 150 seconds, since the total offered load from both the flows exceeds the total available capacity of the link. This approach works even with a number of other contenders and for traffic from the clients in the uplink direction.

Figure 6 shows the efficacy of $A_c$ estimation for WiFi and WiMAX networks, when a client receiving backlogged traffic from the base station moves away from the base station. The top graphs show the end-to-end Iperf throughput, and the bottom graphs show what the estimation functions return.

## 3.4 Adaptive Shaping and Probing

To deploy MESA as a service (potentially on a different node than the base station) and make it independent of the particular wireless technology and the corresponding base station, we make a generic assumption that the base station has a *queue* for incoming packets. The queue is finite in practical implementations, and its size influences the packet delays and packet drops. Hence, one of MESA's goals is to minimize the impact of such queuing in the base station on the video frame delivery. To achieve the same, the shaper dequeues MESA's video buffer at a rate that is *at most* equal to the available wireless channel capacity, thereby minimizing queue buildup on the base station.

The efficacy of available capacity estimation with history based techniques depends on the amount of traffic sent (e.g. see *get_available_capacity* function). Hence, the shaper has to adapt the amount of traffic sent (1) to allow better estimation of available capacity, and (2) not cause buffer overflows at the base station. We achieve these two conflicting goals through

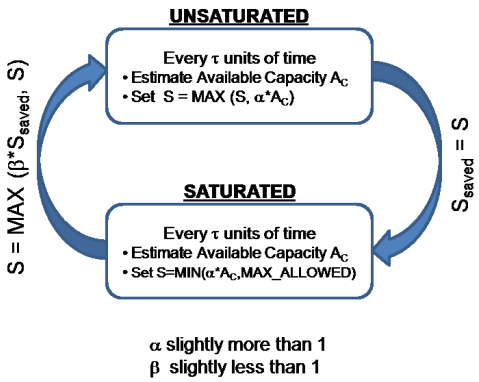



**Figure 7: Adaptive Shaping and Probing.**

a joint probing and shaping approach.

Figure 7 shows the overview of the joint probing and shaping approach. We use the term $S$ to represent the shaper rate at which the video buffer is dequeued. We define two states in which the shaper can be based on whether the wireless channel is at saturation or not, determined by the *at_saturation* function. In each of the two states, we set the shaper rate $S$ as shown in Figure 7. The setting is reevaluated every $\tau$ units of time. When the link is not saturated, $S$ is set to the maximum of (i) current shaper rate $S$ and (ii) the current estimated available capacity $A_c$ multiplied by $\alpha$. Using the maximum of the two helps avoid misprediction of capacity due to fall in video rate itself; in this case we maintain the shaper rate at $S$. With $\alpha > 1$, we achieve a *joint probing* effect that ensures the shaper ramps up (multiplicatively) if more capacity is both available and required.

When the link is saturated, we set the shaper rate to a little more than the available capacity, as long as the maximum allowed rate is not exceeded. In technologies like WiMAX, video traffic is classified as RTPS type, and the RTPS traffic class has a maximum allowed (or sustained) rate for each flow. This restriction should be especially honored in saturation conditions to reduce the effect of flows on each other. In the unsaturated state, however, even video rates higher than `MAX_ALLOWED` may be accepted to maximize the video quality, since spikes in video rates often contain detail or action [21]. For WiFi, we set `MAX_ALLOWED` to the capacity estimated by *get_ett_capacity()*, since it is accurate in the saturated state.

Finally, the shaper can transition from saturated to unsaturated state and vice versa under (1) momentary channel fluctuations due to fading or shadowing, and (2) long-term variation due to mobility, new user arrival, etc. To avoid over-reacting to momentary fluctuations, when the shaper transitions from unsaturated to saturated state, we save the last shaper rate in $S_{saved}$ unsaturated state, and use it to transition back when we move into unsaturated state again. However, we introduce a *hysteresis* parameter $\beta$ to ensure that the shaper does not flip between the saturated and unsaturated states (due to the use of $S_{saved}$) when the capacity is actually reduced. We provide more analysis of the shaper state machine behavior in the Appendix, and also discuss how to choose $\alpha$, $\beta$ and $\tau$.

### 3.5 Frame drop policy

MESA drops complete video frames (i.e. all packets belonging to a video frame chosen for dropping are dropped) in order that the remaining video data rate is less than or equal to the achievable link throughput for each user. Video frames that are considered for rate-matching are dropped in the increasing order of their priority. For example, for H.264 AVC video, B-frames that have lower priority than P-frames are dropped first. If no more B frames exist, P frames are dropped before considering I frames. If multiple frames of the same priority level are available in the buffer, we first drop the frames with the later deadlines, since earlier frames in the GOP are typically used for decoding the later frames and hence are more important.

Working at frame granularity instead of packet granularity makes the solution light-weight, and localize the effect of dropping that minimizes video distortion. Also, the overhead of per-frame granularity rate matching scales with number of frames per second, which changes relatively less frequently than that of per-packet granularity that scales with video rate.

## 4. EXTENSIONS FOR HANDOVER

Today's wireless networks are required to support seamless mobility when users move from one network to another both within the same technology and across technologies. In this direction, many standards (e.g. 802.21 [5] and UMA [15]) and other studies [40] focus on different aspects of such intra- and inter-technology handovers. Since MESA is an in-network solution deployed close to the wireless channel, such handovers can affect the efficacy of MESA. In this section, we briefly consider different scenarios that affect MESA, and discuss the design extensions and alternatives. In particular, the following scenarios arise when moving from one network to another depending on whether the first and second networks have the MESA module in the data path.

**(S1) NO-MESA to MESA-2 and (S2) MESA-1 to MESA-2, oblivious to MESA-1:** In scenario S1, the first network does not have MESA, and the second network does (MESA-2). In S2, both networks have MESA, but MESA-1 is oblivious to a handover when it happens. Handovers of type S2 may happen due to uninformed or unanticipated disruptions to wireless connections, or where the connection is anchored at a node farther from MESA-1 in the network, and hence cannot inform MESA-2 without extra messaging through protocol and/or standard changes. An example of such anchoring is the current UMA-based [15]

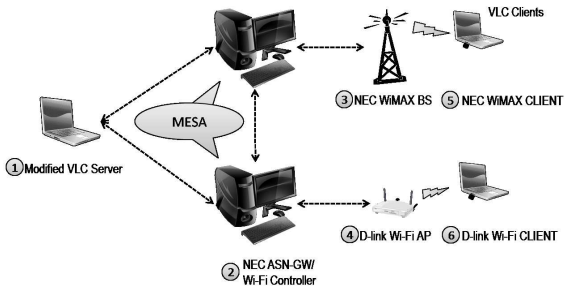



**Figure 8: WiMAX and WiFi prototypes.**

cellular to WiFi handovers used by T-Mobile [14].

Now, recall that the pre-buffering component of MESA delays all packets by $\mathcal{D}$ units of time. In either of the handover scenarios, the delay introduced by the pre-buffering component of MESA-2 can affect the frame delivery and rate matching process. Hence, it seems best to perform minimal pre-buffering in networks that frequently involve handovers, and that do not include messaging to inform MESA of handovers.

**(S3)MESA-1 to MESA-2, awareness at MESA-1:** In this scenario, both networks have MESA, and MESA-1 can be easily informed of a handover. This is possible in all-IP wireless networks, and with WiMAX specifically, where the ASN gateway is aware of every handover and also runs MESA. In this scenario, a simple control message from MESA-1 to MESA-2 informs the value of $\mathcal{D}+\mathcal{R}$ to set deadlines appropriately; further, MESA-2 does not peform pre-buffering.

## 5. PROTOTYPE

We have implemented a prototype of MESA using the Click Framework [34]. MESA is deployed on the NEC ASN gateway and the WiFi controller respectively for the WiMAX and the WiFi testbeds. Both the ASN gateway and the WiFi controller run on Linux machines. The ASN gateway (WiFi controller) is connected to the base station (WiFi AP) with a 100 Mbps link. We implement MESA as a user-level Click module. Using iptables, we let all video flows incoming from the wired side be routed to MESA. Figure 8 shows the prototype setup for both WiMAX and WiFi.

The WiMAX testbed uses an NEC Pasowings mobile WiMAX (802.16e) basestation that transmits in the 2.585-2.595 GHz channel. To enable indoor evaluation, we use a low gain antenna for transmission and also reduce the transmit power of the base station to values that are comparable to WiFi. In the Wifi setup, we use legacy 802.11a/b/g compliant Atheros cards for both the AP and clients with Madwifi Drivers. The Algorithms for capacity estimation are implemented as described in Section 3.3.2. Due to some technical restrictions over *snmp* messaging, we are limited to $\tau = 300$ ms (for the WiMAX experiments) to maintain spacing between successive SNMP queries. For the WiFi testbed, $\tau = 30$ ms. We set $\alpha = 1.04$, and $\beta = 0.92$ based on the analysis in the Appendix.

We use a modified version of VLC [17] for Linux on both server and clients for streaming video. VLC streams a video using fixed packet sizes (default is 1300 bytes) causing certain packets to contain multiple frames. We modified the VLC player to packetize the video frames such that each packet carries only one type of frame. We mark the TOS field in the IP-header of every outgoing packet to indicate the priority level of the frame type it contains. Currently, our prototype assumes live streaming of H.264 AVC videos. Hence, packets belonging to I-,P- and B-frames are marked as 0x1 for I,0x2 for P, 0x3 for the first B after P, and 0x04 for the second B after P, respectively. Voice packets are marked as 0x1. Finally, to maintain backwards-compatibility, MESA forwards unmarked packets to the basestation without any rate matching.

The streaming server runs on an Ubuntu Linux machine. This server is connected to the NEC ASN-GW(or the Wifi Controller) with a 100Mbps wired link. The VLC player on the client side needs no modifications to play the received video stream. However, we added code for reporting the index and type of the frames that were received late, lost or corrupted.

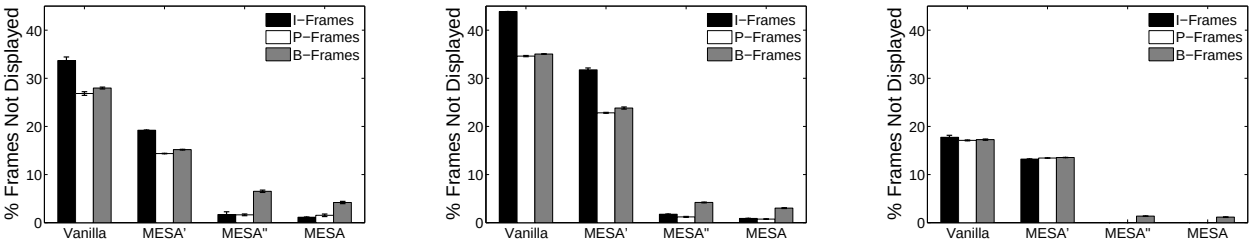
## 6. EVALUATION

In this section, we evaluate in detail the efficacy of the different design decisions we made in MESA on both WiMAX and WiFi platforms. All the measurements are performed in an indoor office environment. In what follows, we present a few representative results from each testbed, and then show the results of a handover from the WiMAX testbed to the WiFi testbed. We experiment with the three videos shown in Table 1.

### 6.1 WiMAX Results

We consider both static and mobile client deployment scenarios for the experiments. WiMAX supports different traffic classes that differ in priority, channel access method and guarantees provided. The class RTPS (Real-Time Polling Service) is suited for video, and the class UGS (Unsolicited Grant Service) is generally recommended for Voice over IP or T1 link emulation. UGS has higher priority than RTPS.

For all video streams, we set up RTPS flows on the WiMAX base station. Since the available capacity at a base station is much larger than what a single video requires, we also setup UGS flows to other WiMAX clients in several of our experiments such that the residual channel capacity for the RTPS flows can be controlled to be around the average rates of the different videos. In a real deployment with many clients, the per-flow provisioning (either at the beginning of the flow setup or due to re-negotiations in response to long-term variations) will typically be a small factor (e.g., 1.2) of the average requirement to maintain good video quality by absorbing most fluctuations and simultaneously maximizing the number of flows admitted.




(a) Trailer Video (b) Sports Video (c) User Video

**Figure 9: MESA's performance with a static WiMAX client. MESA' is a variant of MESA without rate matching and MESA" is a variant of MESA without prebuffering.**

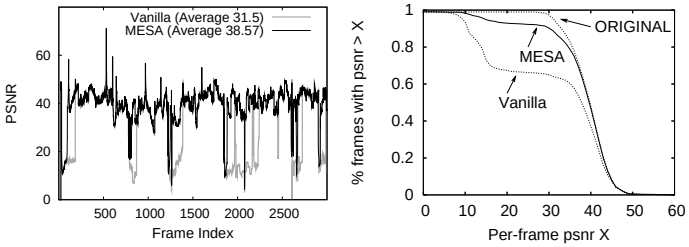
### 6.1.1 Single-Flow Static Experiments

In this experiment, we place a client in a static non-line-of-sight location away from the base station. Using Iperf [2], we found that the good-put of the channel is 3 Mbps on an average. We place another client with a UGS flow such that the residual capacity for the RTPS flow is 1.5 Mbps, 1.4 Mbps and 1.2 Mbps for Trailer, Sports and User videos respectively.

Figure 9 shows the frames not displayed at the receiver with MESA and without MESA (called Vanilla), averaged over five runs, when we stream each of the three videos. The frames not displayed includes frames that were received after their playout deadline, frames that were completely lost and frames that were corrupted due to some packet loss. The graph clearly shows the benefit of MESA over Vanilla. For instance, while the Vanilla case led to a total of about 30% undisplayed frames for the Trailer video, carefully dropping 5% of B-frames with MESA led to <3% of total frames not displayed.

We also plot the results for MESA without the rate-matching component (MESA') and MESA without the pre-buffering component (MESA") to explore their individual benefits. Although MESA' shows improvement in the number of frames not displayed over Vanilla, the results show that it not sufficient as a complete solution. While increasing the initial pre-buffering time $\mathcal{D}$ would show more improvement, it will increase the delay perceived by the user, and may cause annoyance to users with interactive applications like video conferencing. MESA" provides greater improvements over Vanilla, and is close to MESA. We realize that the benefit of a small amount of pre-buffering is more pronounced with videos having greater frame size fluctuation; hence MESA improves more over MESA" with Trailer video that has greater fluctuations than others.

We now demonstrate the benefit of the shaping component. Figure 10(a) shows the benefit of having the adaptive shaper. The top graph shows the video rate fluctuations, the bottom graph shows the number of frames not displayed with MESA, and the middle graph shows the effect of running MESA without the shaper. The middle graph shows increased number of frames not displayed at several time instants due to overestimation of capacity without a shaper that results in




(a) Per-frame (b) CDF

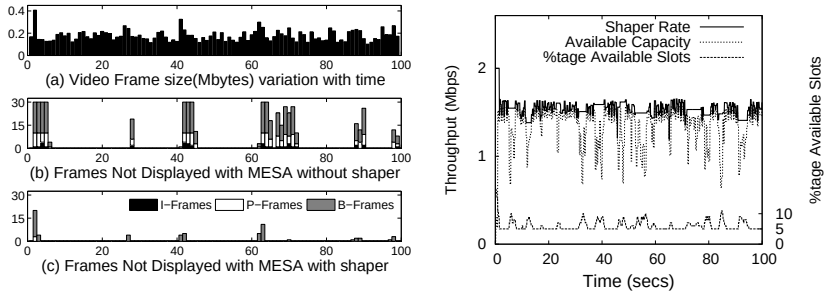
**Figure 11: Per-frame PSNR. The average PSNR for the transmitted video was 40.**

inaccurate rate matching, thereby causing base station buffer overrun. Figure 10(b) shows the behavior of the Shaper with time in controlling the burstiness of incoming traffic and the variation in available capacity for WiMAX. The plot also shows the percentage of available slots in the base station; if available slots are $\leq 5\%$, it indicates that the basetation is in saturation.

Figure 11 plots the per-frame PSNR (peak signal to noise ratio) and its CDF for the Trailer video. PSNR is a standard metric of video quality [28], and is a function of the mean square error between the transmitted video and that displayed at the receiver. We use the EvalVid tool [28] to compute the PSNR. Klaue et al. [28] also show the mapping between PSNR and qualitative user rating; PSNR>37 is considered as excellent quality and PSNR between 31 and 37 is considered good. Figure 11(a) shows that Vanilla leads to significantly greater number of instances of low PSNR, with an average of 31.5. Whereas, MESA improves the PSNR by 7 dB to 38.5. The graph also shows that the fluctuations are short-lived, and hence solutions such as transcoding will not be effective. Figure 11(b) plots the CDF that shows that a user experiences excellent quality more than 90% of the time, as opposed to 60% without MESA. Since the number of frames not displayed directly lowers the PSNR, in the rest of the paper, we use the number of frames not displayed as a metric of evaluation for simplicity.

### 6.1.2 Comparison with WiMAX QoS parameters

In this experiment, we study the efficacy of MESA over setting a WiMAX QoS parameter associated with RTPS flows—`MAX_DELAY`—which allows the base station to




(a) Shaper benefit (b) Shaper behavior

**Figure 10: Shaper benefit and behavior with a static WiMAX client.**

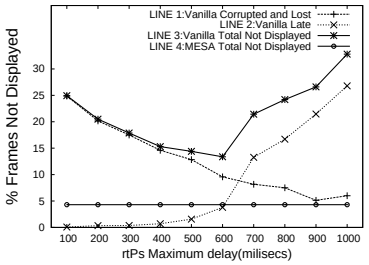
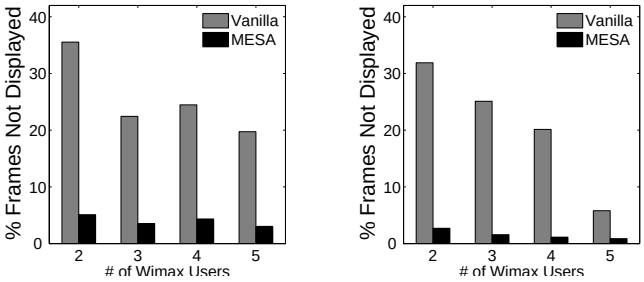



**Figure 12: Effect of setting WiMAX RTPS flow delay parameter on the video quality.**




(a) Trailer Video (b) Sports Video

**Figure 13: Percentage frames not displayed at the receiver with multiple WiMAX users.**

drop packets that are late by a given amount of time. The rationale behind `MAX_DELAY` is also the same as MESA that when the capacity is low, there is no need to transmit frames that will be late at the receiver. Configuring the `MAX_DELAY` parameter, however, makes all packets inherit the same value, does not consider different priority levels of packets, and can corrupt more frames due to packet-wise dropping. This effect can be clearly seen in Figure 12. For this experiment, we vary the `MAX_DELAY` parameter for the Vanilla case and measure the number of frames not displayed. Total frames not displayed (Line 3) is the sum of late frames (Line 2) and corrupted/lost frames (Line 1). Small values of `MAX_DELAY` cause fewer late frames but greater number of corrupt and lost frames due to aggressive dropping of packets, whereas large values cause greater number of late frames than corrupt and lost frames. The difference between Line 3 and Line 4 clearly shows that MESA makes more informed choices leading to significant improvement over even the *best* setting of `MAX_DELAY`. With MESA, we configure the `MAX_DELAY` parameter for each flow to be a large value to avoid its effect.

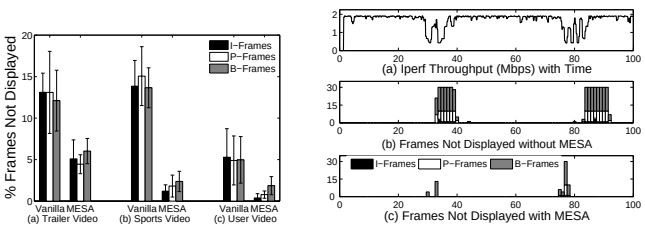
### 6.1.3 Multi-Flow Static Experiments

We now consider the scenario in which multiple clients receive video traffic. We place multiple clients at different static locations in the office cubicles. Similar to the single-flow experiments, we setup a UGS flow to a client to control the residual capacity to the RTPS flows. To restrict the number of parameters that are variable, we use the same video across all RTPS flows in each experiment. We start with one RTPS flow and one UGS flow, and with the addition of every new RTPS flow, we reduce the offered load of the UGS flow by the average rate of the video.

Figure 13 shows for two experiments (with two videos) the average percentage of video frames not displayed with MESA compared to Vanilla for different number of clients. The results are averaged for all flows over five runs. The transmission of video flows from the server to the different clients were staggered by 1 second each to avoid synchronization. The graph shows that MESA achieves significant gains over Vanilla in the multiple flow scenario. While there is some statistical multiplexing gain with increasing number of users, MESA still provides significant benefits. Further, a solution like MESA would allow decreasing per-user provisioning with increasing number of flows to support a greater number of users simultaneously.

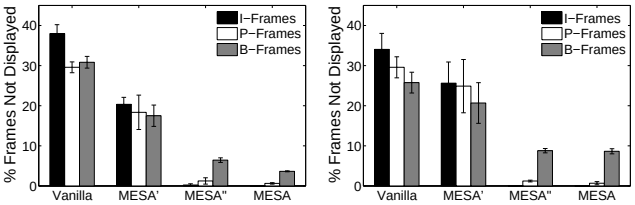
### 6.1.4 Mobility Experiments

Our next experiment includes a mobile client moving along several paths in the corridors of the office. We measure the effect of fluctuating channel conditions due to mobility and the effectiveness of MESA in handling the fluctuations. Figure 14(a) shows the aggregate percentage of frames (averaged over 5 runs) affected with Vanilla and MESA for one path for three different videos. Provisioning is done such that the




(a) Average (b) With time

**Figure 14: With a mobile WiMAX client.**




(a) Trailer Video (b) Sports Video

**Figure 15: Benefit for a Static WiFi client.**

available capacity for the RTPS flow is about 0.5Mbps more than the average when near the base station and about 0.5Mbps less than the average when it is the farthest. The low number of frames getting affected with MESA shows its efficacy in minimizing video distortion. Figure 14(b) shows the results of the same experiment with time for the Sports video. We also plot the throughput obtained by an Iperf flow in the same path in a *different run*; this is why the losses are not synchronized in time. The graph shows that there are regions on the path where the throughput falls down significantly, and MESA has substantial benefits in those regions. For the other regions, available capacity is much higher than required, and hence all frames are displayed. We make similar observations for other paths.

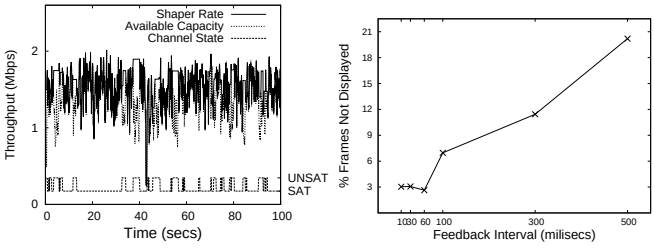
## 6.2 WiFi Results

We repeat several experiments on the WiFi testbed to show that the observations are similar.

### 6.2.1 Static Experiments

This experiment is similar to the one in Section 6.1.1. We place the WiFi client and base station in 802.11g channel 6, and set the bit rate to 6 Mbps to avoid the artifacts of rate adaptation. We also setup a contending constant rate UDP flow between two other WiFi nodes such that the available capacity for the video flow is around the average rate of the different videos. In this setup, there is also uncontrolled interference from the building's operational WiFi networks (although lightly loaded).

Figure 15 shows the frames not displayed at the receiver for two videos, averaged over five runs with and without MESA. We also plot the results for MESA without buffering and MESA without rate matching to explore their individual effects. The benefits of MESA can




(a) (b)

**Figure 16: (a)Behavior of the shaper with WiFi. (b) Sensitivity of MESA to $\tau$.**

be easily observed in the figure. Although the results of Wifi appear similar to those observed in the WiMAX experiments, the error bars in the case of WiFi indicate marked difference between the various runs. This is because of the finer time-scale fluctuations in capacity of a WiFi link as compared to a WiMAX link, due to uncontrolled interference and the behavior of CSMA with contending flows.

We now study a multi-flow case where a MESA flow operates in conjunction with a legacy video flow (which causes significant fluctuation in available capacity for MESA). We consider a setup in which we stream videos from two base stations to two clients on different networks but operating on the same channel. In the first run, both videos were streamed without MESA. The percentage of video frames not displayed for Videos I and II was 22% and 12% respectively. In the second run, video I was streamed using MESA, and Video II was streamed without MESA. The percentage of frames not displayed reduced to 1.73% for Video I and remained close to 12% for Video II, thereby demonstrating the efficacy of MESA even with significant fluctuation caused by a contending flow.

### 6.2.2 Sensitivity to $\tau$

Figure 16(a) shows the behavior of the shaper with time. Unlike WiMAX observations in Figure 10(b), the shaper rate fluctuates significantly in WiFi due to our choice of $\tau$; $\tau$=30ms for WiFi and $\tau$=300ms for WiMAX. To better understand the impact ot $\tau$ on the displayed video, we consider the above static setup again with one client receiving video, and plot in Figure 16(b) the total number of frames not displayed when varying the value of $\tau$. The graph clearly shows the benefit of smaller values of $\tau$, although the shaper rate fluctuations increase at smaller values of $\tau$. However, very small values also add significant feedback overhead between MESA and the base station for little extra benefit. This tradeoff should be taken into consideration for setting the parameters, when deploying MESA in a real setting.

## 6.3 WiMAX–WiFi Handover

In this experiment, we perform a network-initiated WiMAX to WiFi Handover emulating the scenario S3 discussed in Section 4. The other scenarios are straight-

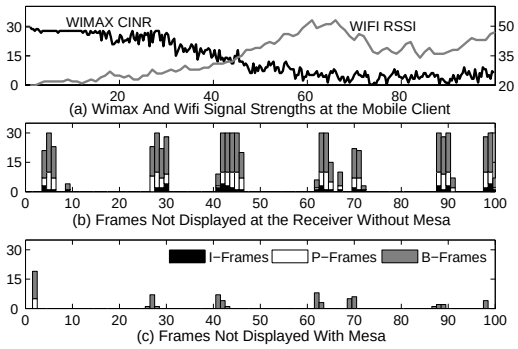



**Figure 17: Handover with MESA.**

forward, and hence we do not show the results here. For emulating scenario S3, we have implemented a simple mobility manager on the ASN Gateway that triggers a handover to WiFi when the CINR for the client falls below a threshold $T$ dB. Once the handover has been triggered, the anchor MESA running on the ASN Gateway sends a control packet containing the value $\mathcal{D} + \mathcal{R}$ to the target MESA running on the WiFi Controller. The target MESA uses the above value for deadline calculation, but does not perform additional pre-buffering. All the current and future video packets from the anchor MESA are then forwarded to the target MESA. The ASN Gateway and the WiFi Controller are connected using the wired network infrastructure of our office. We place the WiMAX and WiFi base stations at two different places in the office. The dual mode mobile client with a WiMAX and a WiFi interface is moved along a path in the office corridor to move away from the WiMAX base station and towards the WiFi base station. Figure 17 shows the results for two runs, once without MESA and once with MESA. The handover from WiMAX to WiFi takes place at around 50 seconds. The graph shows that MESA ensures a smooth video delivery by dropping low priority frames at precise time instants.

## 7. DISCUSSION AND CONCLUSION

We describe the design and instantiation of a multimedia edge service architecture (MESA) for enhanced video transmission over wireless networks. The key idea is to match the video rate to the instantaneous available wireless channel capacity, thereby adapting to variations induced by channel fading, shadowing, user mobility, etc. at fine timescales. The need and effectiveness of rate matching depends on a number of other factors such as admission control, the extent to which over-provisioning is done to achieve statistical multiplexing, the workload at any instant of time, etc. The benefit of MESA comes when a momentary fluctuation in available capacity due to any such condition can degrade the video quality. MESA makes the quality degrade more gracefully. Finally, performing rate matching as a service at the wired-wireless edge (transition point) achieves high accuracy, facilitates immediate deployment, and enables a design applicable to several wireless technologies, with the augmentation of a few technology-specific mechanisms.

MESA is an attractive solution for both MNOs and the emerging MVNOs who strive to provide enhanced services to differentiate themselves from other MVNOs and MNOs in the market [4, 26, 30]. Although we focus on downlink in this paper, MESA can also be easily adapted to run on mobile devices for live streaming [13] or video conferencing in the uplink direction.

## 8. REFERENCES


[1] Beceem wimax chips for mobile applications. http://www.beceem.com/.
[2] Iperf. *http://sourceforge.net/projects/iperf/*.
[3] Linphone, an open-source SIP video phone for Linux and Windows. http://www.linphone.org/.
[4] List of mvnos and their specializations. *http://www.mobileisgood.com/mvno.php*.
[5] Media Independent Handover. http://www.ieee802.org/21/.
[6] Mobile wimax solutions from nec. http://www.nec.com/global/solutions/nsp/WiMAX/.
[7] Movik networks content-aware mobile edge. http://www.movik.net/solutions/deliver.html.
[8] Nec access network solutions for telecom operators. http://www.nec.com/global/solutions/nsp/accessnetwork/index.html.
[9] Nvidia concept phone with 720p hd video. http://mobilementalism.com/2008/02/14/nvidia-concept-phone-wows-the-mobile-world-with-720p-hd-video/.
[10] Ortiva wireless. http://www.ortivawireless.com/.
[11] Overview of 4g systems. *http://en.wikipedia.org/wiki/4G*.
[12] Picochip femtocell solutions. http://www.picochip.com/.
[13] Qik: Share live video from your mobile phone. http://qik.com.
[14] T-Mobile Hotspot@home. http://www.theonlyphoneyouneed.com/.
[15] Unlicensed Mobile Access: technology Overview. http://www.umatechnology.org/overview/.
[16] Video Calls and Video Conferencing with Skype. http://www.skype.com/allfeatures/videocall/.
[17] Vlc media player. *http://www.videolan.org/vlc/*.
[18] Meru's first video-over-wireless infrastructure solution. http://www.merunetworks.com/news/press_releases/index.php?articleID=041409, April 2009.
[19] M. Burza, J. Kang, and P. V. D. Stok. Adaptive streaming of mpeg-based audio/video content over wireless networks. *Multimedia*, 2(2), April 2007.
[20] H.-C. Chuang, C. Y. Huang, and T. Chiang. A novel adaptive video playout control for video streaming over mobile cellular environment. In *Circuits and Systems, IEEE Symposium on*, May 2005.
[21] W.-C. Feng and J. Rexford. Performance evaluation of smoothing algorithms for transmitting prerecorded variable-bit-rate video. *Multimedia, IEEE Transactions on*, 1(3):302–312, Sep 1999.
[22] M. Grossglauser, S. Keshav, and D. N. C. Tse. Rcbr: a simple and efficient service for multiple time-scale traffic. *IEEE/ACM Trans. Netw.*, 5(6):741–755, 1997.
[23] Z. Jiang and L. Kleinrock. A general optimal video smoothing algorithm. In *IEEE INFOCOM*, 1998.
[24] H. Juan, H. Haung, C. Huang, and T. Chian. Scalable video streaming over mobile wimax. In *Proc. IEEE International Symposium on Circuits and Systems*, 2007.
[25] M. Kalman and B. Girod. Optimized transcoding rate selection and packet scheduling for transmitting multiple video streams over a shared channel. In *Proc. IEEE International Conference on Image Processing*, 2005.

[26] A. Kiiski. Impacts of mvnos on mobile data service market. In *Euro. Regional ITS Conference*, 2006.
[27] A. Kim, H. Nam, S. Lee, and S. Ko. Probing-based channel adaptive video streaming for wireless 3g network. In *Proc IEICE Transactions on Communications*, 2006.
[28] J. Klaue, B. Rathke, and A. Wolisz. Evalvid - a framework for video transmission and quality evaluation. In *Int. Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, 2003.
[29] T. Lakshman, A. Ortega, and A. Reibman. Vbr video: tradeoffs and potentials. *Proceedings of the IEEE*, 86(5):952–973, May 1998.
[30] G. Lenahan. With the right support, mvnos can enrich network operators with innovation differentiati on and market share. Telcordia whitepaper: *http://telcordia.com/library/whitepapers/mvno_mvne.jsp*.
[31] D. Levine, I. Akyildiz, and M. Naghshineh. A resource estimation and call admission algorithm for wireless multimedia networks using the shadow cluster concept. *IEEE/ACM Trans. Networking*, 5(1):1–12, Feb 1997.
[32] G. Liebl, T. Schierl, T. Wiegand, and T. Stockhammer. Advanced wireless multiuser video streaming using the scalable video coding extensions of h.264/mpeg4-avc. In *IEEE ICME*, 2006.
[33] M. Lu, P. Steenkiste, and T. Chen. Video streaming over 802.11 wlan with content-aware adaptive retry. In *Proc. IEEE Int. Conference on Multimedia and Expo*, 2007.
[34] R. Morris, E. Kohler, J. Jannotti, and M. F. Kaashoek. The click modular router. *SIGOPS Oper. Syst. Rev.*, 33(5):217–231, 1999.
[35] K. Ramachandran, R. Kokku, H. Zhang, and M. Gruteser. Symphony: synchronous two-phase rate and power control in 802.11 wlans. In *ACM MobiSys*, 2008.
[36] J. D. Salehi, S.-L. Zhang, J. Kurose, and D. Towsley. Supporting stored video: reducing rate variability and end-to-end resource requirements through optimal smoothing. *IEEE/ACM Trans. Netw.*, 6(4):397–410, 1998.
[37] H. Schwarz, D. Marpe, and T. Wiegand. Overview of the scalable video coding extension of the h.264/avc standard. *Circuits and Systems for Video Technology, IEEE Transactions on*, 17(9):1103–1120, Sept. 2007.
[38] B. Shen, W. Tan, and F. Huve. Dynamic video transcoding in mobile environments. In *Proc. IEEE Multimedia*, 2008.
[39] H. Song and D.-B. Lee. Effective quality-of-service renegotiating schemes for streaming video. *EURASIP J. Appl. Signal Process.*, 2004:280–289, 2004.
[40] A. H. Zahran, B. Liang, and A. Saleh. Signal threshold adaptation for vertical handoff in heterogeneous wireless networks. *Mob. Netw. Appl.*, 11(4):625–640, 2006.
[41] H. Zhang, Y. Zheng, M. A. Khojastepour, and S. Rangarajan. Scalable Video Streaming over Fading Wireless Channels. In *WCNC*, 2009.
[42] Q. Zhang, W. Zhu, and Y.-Q. Zhang. Channel-adaptive resource allocation for scalable video transmission over 3g wireless network. *IEEE Trans. Circuits and Systems for Video Technology*, 14(8):1049–1063, Aug. 2004.

## Appendix

In this section, we provide a brief analysis of the adaptive shaper design. Specifically, we discuss the interplay between the parameters $\alpha$, $\beta$ and $\tau$. We consider four canonical cases of capacity changes, and demonstrate the various properties that make the shaper effective in adapting to the capacity changes. Figure 18 shows the four cases of capacity changes with time. $T1$ and $T2$ represent capacities before and after the change respectively in each case with time. U and S represent that the base station is in unsaturated and saturated states respectively. Cases (b), (c) and (d) include a momentary fluctuation that brings the link capacity significantly below $T1$.

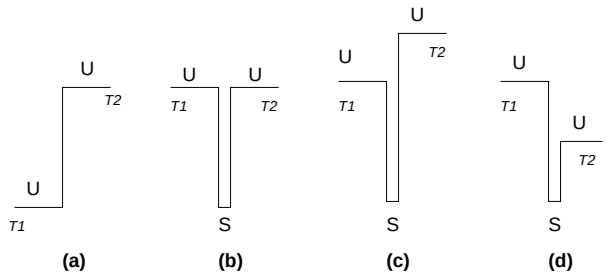



**Figure 18: Shaper Cases. U represents the unsaturated state and S represents the saturate state. T1 and T2 are the correct available rates to which the shaper rate should converge.**

**Case (a):** In this case, $T2 > T1$. Let $t$ be the amount of time for the shaper rate to reach $T2$ from $T1$. In the unsaturated state, the shaper rate increases by a multiplicative factor of $\alpha$ every $\tau$ units of time. Hence, it takes $t/\tau$ rounds to reach $T2$. Let $f = T2/T1$ represent the factor by which the capacity changes. Then,

$$T1 * \alpha^{t/\tau} = T2 \quad (3)$$

$$\Rightarrow t/\tau = \frac{lnf}{ln\alpha} \quad (4)$$

Figure 19(i) shows the plot of $\alpha$ and the number of rounds $t/\tau$ for different values of $f$. From the graph, we can see that with $\alpha$ =1.04, it takes less than 5 rounds for the shaper to catch up with a 10–20% increase in capacity, and less than 20 rounds for doubled capacity. As an example, with $\tau$ =30ms, even if capacity doubles, the shaper takes just about 500ms to ramp up to the right value.

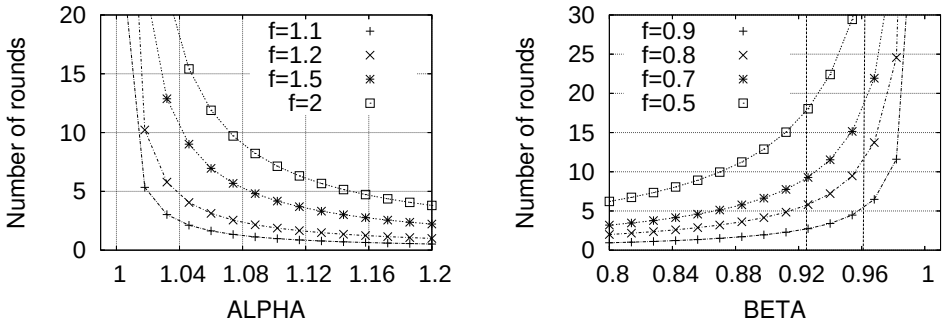
**Case (b):** In this case, $T1 = T2$. However, a momentary fluctuation takes the shaper into the saturated state. When the system moves out of the saturated state, the shaper state is initially set to $T1 * \beta$. For this value to reach $T1$, if it takes t units of time,

$$T1 * \beta * \alpha^{t/\tau} = T1 \quad (5)$$

$$\Rightarrow t/\tau = -\frac{ln\beta}{ln\alpha} \quad (6)$$

Setting $\beta = 1/\alpha$ ensures that the number of rounds is 1, i.e. it would take just one measurement interval to return to the correct capacity after a momentary fluctuation due to fading or shadowing. Setting $\beta = 1/\alpha^2$ increases the number of rounds to 2.

**Case (c):** In this case, $T2 > T1$. This case is different from case (a) in that it considers a momentary fluctuation that takes the shaper into saturated state. When the shaper gets out of the saturated state, it starts at




(i) (ii)

**Figure 19: (i) Sensitivity to Alpha (Case (a)), (ii) Sensitivity to Beta (Case (d)).**

$T1 * \beta$. For this value to reach T2,

$$T1 * \beta * \alpha^{t/\tau} = T2 \quad (7)$$

$$\Rightarrow t/\tau = \frac{lnf - ln\beta}{ln\alpha} \quad (8)$$

Here again, setting $\beta = 1/\alpha^2$ instead of $\beta = 1/\alpha$ increases the number of rounds by 1.

**Case (d):** In this case, $T2 < T1$. This case considers the scenarios when the base station goes into saturated state for all shaper rates greater than $T2$. Here, the shaper essentially cycles through a series of transitions between unsaturated and saturated states to reach the final value $T2$ from $T1$. For every cycle from unsaturated to saturated to unsaturated, the shaper rate reduces by a factor $\beta$. Every such cycle takes $2\tau$ units of time. Hence, if $t$ is the time taken for the shaper to reach $T2$ from $T1$, then

$$T1 * \beta^{t/2\tau} = T2 \quad (9)$$

$$\Rightarrow t/\tau = \frac{2lnf}{ln\beta} \quad (10)$$

Figure 19(ii) plots the variation of $t/\tau$ with $\beta$ for different values of $f$. Further, we draw two vertical lines at $\beta = 1/\alpha$ and $\beta = 1/\alpha^2$, when $\alpha = 1.04$. The graph clearly shows that setting $\beta$ at $1/\alpha^2$ leads to significantly fewer rounds, especially when the drop in capacity is high.

From cases (b)–(d), setting $\beta = 1/\alpha^2$ provides a good tradeoff for covering all cases, when $\alpha = 1.04$.