Real-Time Video Streaming using H.264 Scalable Video Coding (SVC) in Multihomed Mobile Networks: A Testbed Approach

Conference Paper · February 2011

DOI: 10.1117/12.872111

3 authors:

James Nightingale
University of Strathclyde

Qi Wang
University of the West of Scotland

Christos Grecos
Central Washington University

Real-Time Video Streaming using H.264 Scalable Video Coding (SVC) in Multihomed Mobile Networks: A Testbed Approach

James Nightingale*, Qi Wang, Christos Grecos
Audio Visual Communications and Networks Group, School of Computing,
University of the West of Scotland, Paisley, United Kingdom

ABSTRACT

Users of the next generation wireless paradigm known as multihomed mobile networks expect satisfactory quality of service (QoS) when accessing streamed multimedia content. The recent H.264 Scalable Video Coding (SVC) extension to the Advanced Video Coding standard (AVC), offers the facility to adapt real-time video streams in response to the dynamic conditions of multiple network paths encountered in multihomed wireless mobile networks. Nevertheless, pre-existing streaming algorithms were mainly proposed for AVC delivery over multipath wired networks and were evaluated by software simulation. This paper introduces a practical, hardware-based testbed upon which we implement and evaluate real-time H.264 SVC streaming algorithms in a realistic multihomed wireless mobile networks environment. We propose an optimised streaming algorithm with multi-fold technical contributions. Firstly, we extended the AVC packet prioritisation schemes to reflect the three-dimensional granularity of SVC. Secondly, we designed a mechanism for evaluating the effects of different streamer 'read ahead window' sizes on real-time performance. Thirdly, we took account of the previously unconsidered path switching and mobile networks tunnelling overheads encountered in real-world deployments. Finally, we implemented a path condition monitoring and reporting scheme to facilitate the intelligent path switching. The proposed system has been experimentally shown to offer a significant improvement in PSNR of the received stream compared with representative existing algorithms.

Keywords: Video streaming, H.264 SVC, testbed, multihomed, mobile networks

1. INTRODUCTION

Pervasive deployment of various wideband wireless technologies provides the infrastructure required to support an emerging wireless paradigm known as multihomed mobile networks, in which a group of users moves together in unison (e.g. on a train or aircraft) and can access to multiple networks simultaneously. Users within these multihomed mobile networks expect satisfactory quality of service (QoS) when accessing streamed multimedia content. Recent standardisation of the H.264 Scalability Video Coding (SVC)[1], a sophisticated extension to the Advanced Video Coding standard (AVC)[2], offers the facility to adapt real-time video streams in response to the dynamic conditions of multiple network paths encountered in multihomed mobile networks. The dynamic nature of these networks provides a challenging environment for real time streaming of multimedia content. The characteristics of each of the available network paths, is time varying as a result of changing conditions within the core network and the two different wireless networks (access & mobile) which comprise the end to end path from streaming server to mobile network client node. Within the mobile network, wireless path conditions vary over time as nodes join or leave the network. Additionally as the vehicle (mobile network) moves and changes its point(s) of attachment to the wider infrastructure, the nature of the access paths to the mobile network vary in terms of delay, available bandwidth and the radio access technology employed. Network layer adaption of H.264 SVC streams provides a promising way of delivering scalable video streams in such dynamic network conditions. H.264 SVC streams extend the limited scalability offered by MPEG4 Fine Grain Scalability (FGS), providing the ability to intelligently drop sub streams (entirely or partially) in response to varying network path conditions. Pre-existing work has proposed scheduling algorithms to address the delivery of H.264 AVC streams over multipath wired networks and presented results obtained from software simulations. The aim of these algorithms is to maximise the number of video packets arriving at the client in time to be of use in the decoding process, while minimising the need for packet reordering at the client. This was achieved by making efficient use of all available network paths.

*james.nightingale@uws.ac.uk

Our work significantly extends previous research, by adapting representative algorithms to SVC, and practically implementing them on a functional, hardware-based multihomed mobile networks testbed. This approach allows us to demonstrate and measure real-time algorithm performance in a realistic environment. We have implemented these algorithms in Linux user space, experimentally evaluated their effectiveness in the context of multihomed mobile networks, and proposed an algorithm optimised for use in when streaming SVC content in this context. In this work we introduce both a realistic testbed and streaming framework for SVC over multihomed mobile networks and a novel path selection and scheduling algorithm optimised for use in mobile networks. This paper is organized as follows. Section 2 presents related work on multipath streaming algorithms and real time delivery of SVC streams. Section 3 describes our proposed algorithm. Section 4 describes our testbed and Section 5 the implementation of representative algorithms. In section 6 we present the results of our testbed experiments and finally section 7 concludes the paper.

2. RELATED WORK

2.1 Mobile networks

Work on the development of mobility protocols for portable devices, which began with the introduction of Mobile Internet Protocol[3] (MIP) and Mobile IPv6[4] (MIPv6), has been extended by the introduction of the Network Mobility protocol[5] (NEMO) in which a group of mobile users move together in unison (e.g. on a bus, train or plane). In NEMO users no longer connect directly to their ISP with their mobile devices, but instead connect to a Mobile Router (MR) on board the vehicle. The MR then acts as the single point of attachment to the wider infrastructure, providing mobility support for all nodes in the mobile network. In multihomed mobile networks, the MR is equipped with multiple network interfaces and can simultaneously connect to and distribute application flows across all available network paths. It makes concurrent use of radio access networks from different service providers and of different (heterogeneous) wireless technologies. Communications between a mobile network node (MNN) and a correspondent node (CN) travel through a bidirectional communication tunnel between the two NEMO mobility management agents, one of which runs in the MR and the other at the Home Agent (HA) situated in the MR’s home network. Packets destined for an MNN are firstly directed to the HA where they are encapsulated with the address of the MR and transmitted through the tunnel. At the MR the encapsulation header is removed and the packet forwarded to the MNN. Both MIPv6 and NEMO permit a device to have multiple care of addresses (CoA), thus allowing an MR to make simultaneous use of multiple tunnels to the HA. There are some factors which can limit the performance of NEMO. One of which is the addition of tunneling overheads to each packet transmitted from CN to MNN and another is the fact that the link from the MR to the access network (particularly in mobile networks which are not multihomed) can become a bottleneck in the path from CN to MNN. Figure 1 shows the network topology of a multihomed mobile network.

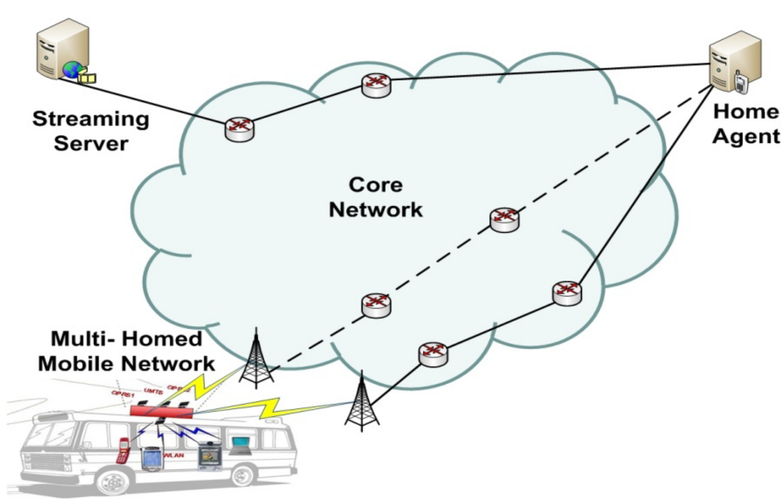

Figure 1. Typical topology of a multihomed mobile network.

In previous work[6,7] we have discussed the problems associated with providing quality of service (QoS) for application flows in multihomed mobile networks and proposed a policy driven QoS aware scheme to provide an 'always best connected path' between CN and MNN. Network metrics and application specific rule sets are used to determine the current 'best path' for an application flow and paths are switched as required in response to changes in network conditions, thus maintaining the always best connected principle

2.2 H.264 Scalable Video Coding.

Recent standardization of the scalability extension (H.264 SVC) of the H.264 Advanced Video Coding (AVC) standard allows the encoding of video sequences as a number of sub-streams. In SVC a stream consists of an AVC compliant base layer, providing a minimum quality of video, and a number of enhancement layers which improve the quality of the received stream. The three dimensional scalability of SVC, which can be used for network or terminal adaption of streams, utilizes picture resolution, frame rate and signal to noise ratio to provide spatial, temporal and quality enhancement dimensions. In order to safeguard valuable network resources, a sender may only send those layers which a client node is capable of processing. When the network is congested or there is insufficient bandwidth to deliver the entire stream, network adaption may drop higher enhancement layers to reduce the bandwidth requirement and ensure the delivery of the base layer and lower enhancement layers. Sacrificing the upper enhancement layers in this manner, provides the user with an acceptable (albeit lower quality) video and makes efficient use of available bandwidth. In the Internet Engineering Task Force (IETF), work to standardize a payload format for delivery of video streams over the widely used Real Time Protocol (RTP) has produced RFC 3984[8], a standard for the RTP payload of H.264 AVC streams. Further work[9] which extends RFC 3984 to H.264 SVC is currently at an advanced stage, but has not as yet been standardized. This proposal describes the payload format of SVC streams for delivery over RTP. A number of options are specified including single Network Abstraction Layer (NAL) units per RTP packet, several strategies for the aggregation of multiple NAL units in an RTP packet and a means of fragmenting a NAL unit over several RTP packets. Despite the development of a limited SVC streaming and playback capability in the Astrals project[10], there are currently no freely available streaming servers or playback clients for SVC. In the absence of such standards and tools, we explore and further develop the Scalable Video Evaluation Framework[11] (SVEF) for the empirical evaluation of real- time SVC streaming.

2.3 Multipath video streaming.

Much previous research has been conducted into the improvement of streaming mechanisms for both single and multipath environments and with respect to wireless path links. However, to the best of our knowledge, this work is the first to provide an empirical investigation of scalable video streaming in a realistic multihomed mobile networks environment. A number of these previous research projects have aimed to improve performance in delay sensitive multimedia streaming applications, some of which have focused on provision of end to end control or QoS mechanisms,[12,13] while others have addressed the issues of efficient use of the aggregated bandwidth of multiple network paths. Some works[14,15,16] have considered the path selection and packet scheduling problem from a media aware perspective. The more recent of these schemes[15,16] have recognized that packets in a video stream have varying degrees of importance in terms of their contribution to the perceived quality of the received video, for example the loss of an 'I' frame would have a greater impact than the loss of a 'B' frame. Multi dimensional scalability in H.264 SVC video streams, creates more complex packet dependencies than those encountered in previous encoding schemes, but also offers the opportunity to introduce more sophisticated packet prioritization schemes with a finer granularity than previous MPEG\AVC based schemes[15,16]. A video packet can only be of use in the decoding process if it arrives at the client before its decoding deadline and all of the packets upon which it depends have previously been decoded. Any path selection and scheduling scheme must, therefore, not only consider both network path characteristics of available bandwidth and delay, but also media packet metrics of relative importance, decoding deadline and successful scheduling of ancestor packets. Additionally the use of multiple simultaneous delivery paths may lead to packets arriving out of sequence at the client and thus requiring an increase in input buffer size and additional processing time to reorder them as they enter the decoding process, this may place a high burden on resource constrained mobile devices.

Chebrolu & Rao addressed the issue of out of order delivery of media packets by proposing Earliest Delivery First (EDF)[14]; a media aware multi-path scheduling scheme. EDF has two components. The first of which is an interface selection mechanism at the mobile client. Each interface is assigned a cost based on operator charges. The minimum number of interfaces required to provide sufficient bandwidth for the streamed media content are selected based on a combination of interface cost and available bandwidth on the associated network paths.

The aggregated bandwidth of all selected paths is then used to deliver the media content by the second component, a scheduling mechanism situated within a mobile node's home network. The arrival time of a media packet is calculated for each available path and the packet is sent on the path which offers the earliest arrival time. The burden of collecting network path statistics (by some unspecified mechanism) is placed on the mobile node which then communicates this data to the scheduling agent within its home network. This proposal applies to a single multi-homed mobile host rather than a mobile network node and considers the overall path characteristics of end to end delay and available bandwidth of each path. It is assumed that the scheduling agent only offers a single client facing network interface with the core network taking care of routing to the mobile node (via selected operator networks). EDF assumes that the aggregated bandwidth is sufficient to deliver the entire video stream. In later work by the same authors the Earliest Delivery Path First (EDPF) algorithm updates EDF by removing the cost based interface selection. The authors address the unequal importance of video packets by proposing a selective frame discard mechanism to drop lower priority packets when insufficient bandwidth is available to deliver the entire stream. Each of these proposals assumes a Quality of Service (QoS) negotiated bandwidth on each path which remains stable for the duration of the streaming session, however, a bandwidth aware dynamic QoS negation mechanism[17] (TS-EDPF) is proposed by Fernandez et al which further extends EDPF by adding a time slot policy. These EDF based proposals only consider non scalable MPEG encoded video streams.

Jurca and Frossard[16] consider the packet selection and scheduling problem of delivering a generic scalable video format over multiple wired network paths. They propose both a theoretical optimal multipath streaming model and a more practical heuristic load balancing algorithm (LBA) which, while similar to EDPF with selective frame discard, gives greater consideration to the effect of complex packet/frame dependencies. Their model assumes firstly that the video streaming server is equipped with multiple interfaces each providing an independent network path to the receiving client and secondly that all scheduling and path selection takes place at the streaming server. This differs from EDF variants where the scheduling agent resides within a mobile client's home network. Time varying network path characteristics are reported to the scheduling process by an (unspecified) network overlay monitoring facility. This proposal, while providing a more sophisticated packet prioritisation scheme than EDF variants, does not consider the multi-dimensional scalability offered by H.264 SVC and assumes that transmission takes place over lossless wired connections.

None of the above schemes have, to the best of our knowledge, been empirically investigated in the context of mobile networks or specifically adapted for H.264 SVC encoded video streams. In this work we implement the EDPF and LBA algorithms on our testbed and compare them to our proposed optimised algorithm. Our practical Linux user space implementations are discussed in section 5.

3. OPTIMISED ALGORITHM

In this section we propose an optimised algorithm which takes account of the previously unconsidered practical factors encountered in both SVC and multihomed mobile networks. As the initial stage, we calculate packet delivery times at the client based on the full network size of a packet including all added overheads rather than the payload size which was previously considered in EDPF and LBA. In mobile networks, there can be multiple levels of tunnelling (and thus added tunnel overheads). For example, if a passenger on a bus carries several portable devices, one of which (e.g. mobile phone) connects to the mobile router on the bus and then shares that connection with his other devices thus creating a personal area network (PAN). Such a scenario would incur an additional level of tunnelling overhead which is not considered in other published work. SVC encoded sequences have a variable bit rate (VBR) and NAL (network abstraction layer) unit sizes which vary significantly. Many unit sizes are larger than the 1500 byte Ethernet Maximum Transmission Unit (MTU). In our 30 fps sample sequences the proportion of RTP packets exceeding the MTU ranged from 58% (*Bus*) to 70% (*Soccer*). These larger RTP packets are fragmented into 'network sized' IP packets. As tunnelling overheads are added to each IP packet, a significant additional overhead can be accrued. For instance, in the 'Soccer' sequence at 30fps additional overheads added 4.83% to the final size of the stream raising the bandwidth requirement from 2.86MBps to 3.01Mbps. These overheads, if not considered, lead to imprecise calculation of packet delivery times at the client and the possibility of incorrect scheduling decisions. We also consider the path switching delay in multihomed mobile networks and seek to reduce the path switching frequency in order to minimize the accrued overhead. Since an RTP packet may contain a number of IP packets and each individual packet represents a potential switching operation; we reduce the number of potential switching operations by scheduling at an RTP packet level rather than at IP packet level. Figure 2 illustrates the RTP packet size distribution for the *Soccer* sequence at 30 fps.

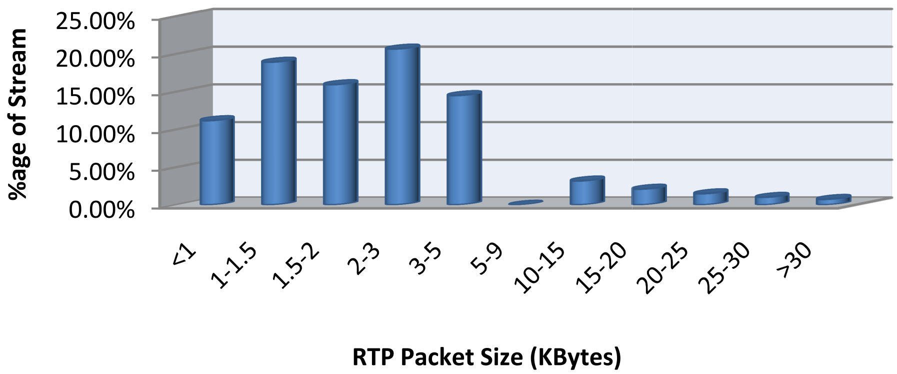

Fig. 2 Packet sizes for *Soccer* sequence at 30fps

A trade off between optimal bandwidth aggregation and path switching frequency is also made by firstly checking if a packet will arrive at the client before its decoding deadline using the current path and only switching paths if a packet will arrive too late to be of use on the current path. This is further justified by the fact that priority weighting and decode deadline data is carried at the RTP/NAL unit level in SVC. It is also noted that the Internet draft[9] for transporting SVC streams over RTP does not provide payload priority information at the IP level. As a further enhancement to previously proposed schemes, in our framework we enhance the LBA packet prioritization scheme in the SVC context and provide an ancestor scheduling check, which determines a packet's ancestors using frame number, scalability information contained in the NAL unit. Only those packets whose ancestors (within the current window of knowledge for real-time streaming) have been successfully scheduled are considered for scheduling. The packets within each read-ahead window are sorted according to packet priority, ensuring that base layer packets will not be dropped in favour of higher enhancement layer packets and each packet is then sent on the path offering the earliest delivery time. As with EDPF, this limits the need for packet reordering at the client.

A diagrammatic comparison of EDPF, LBA and our optimised algorithm is shown in Figure 3, listing the inputs considered by each algorithm and the possible scheduling outcomes. Figure 4 provides a stepwise description of our algorithm, with its practical implementation being described in section 5.

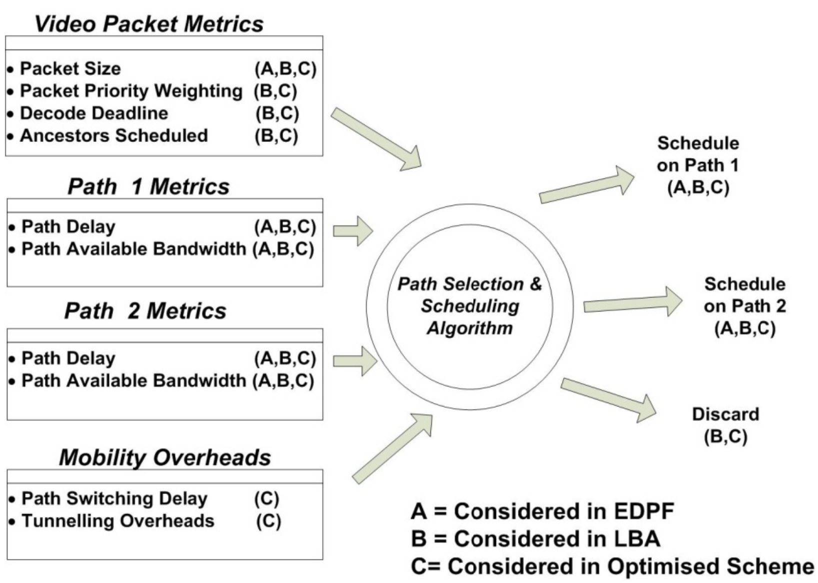

Fig. 3 Factors used by scheduling algorithms

```
100 Sort packets in window by weighting and decode deadline
110 For each packet in fetch window:
120   If ancestors (in current window) have been scheduled
130     Calculate full network size of packet
140     Calculate arrival time (at client) on current path
150     If arrival time within decode window
160       make current path candidate path
170     Else
180       For each alternative path
190         Calculate arrival time (at client) on this path
200         If arrives within decode window and offers
            fastest arrival time make this path candidate path
210       Next Path
220     End If
230     If a candidate path is found
240       If candidate path <> current path
250         switch path, current path = candidate path
260       End If
270       If path switched and  would arrive on new pathbefore decode window opens
           wait till arrival time = start of window
280        Transmit packet on current path
290        Update available time of current path
300     Else
310       Discard packet, record fail point
320     End If
330   Else
340     Discard Packet, record fail point
350   End If
360 Next Packet
```

Fig. 4 The pseudo-code of optimised algorithm

4. TESTBED

Our multihomed mobile networks testbed consists of standard PCs running Ubuntu Linux for the video streaming server, the mobility management agent, the core routers, the mobile network router and mobile network clients in our testbed. Two paths are provided between the video streaming server and the multihomed mobile network. Each path consists of a 100Mbps Ethernet wired link incorporating a core router running wide-area network emulation and path monitoring modules and an IEEE 802.11g wireless link offered by a modified Linksys WRT54GL wireless router. All PC's used in the testbed have 3.4 GHz Pentium 4 processors with mobility agents and core routers having 1 GB of RAM and the end nodes (streaming server and mobile client) having 512 MB of RAM. A topological diagram of the testbed is shown in Figure 3. Mobility management is provided by open source Mobile Internet Protocol version 6 (MIPv6) based software running at both the Home Agent (HA) and the Mobile Router (MR). In the design of our testbed, we incorporate mechanisms to provide path switching and path monitoring functions which, when combined with our scheduling implementation, provide an application specific (SVC streaming) instance of the Network Selection Algorithm[6] (NSA) or Common Radio Resource Manager[7] (CRRM) introduced in our previous work.

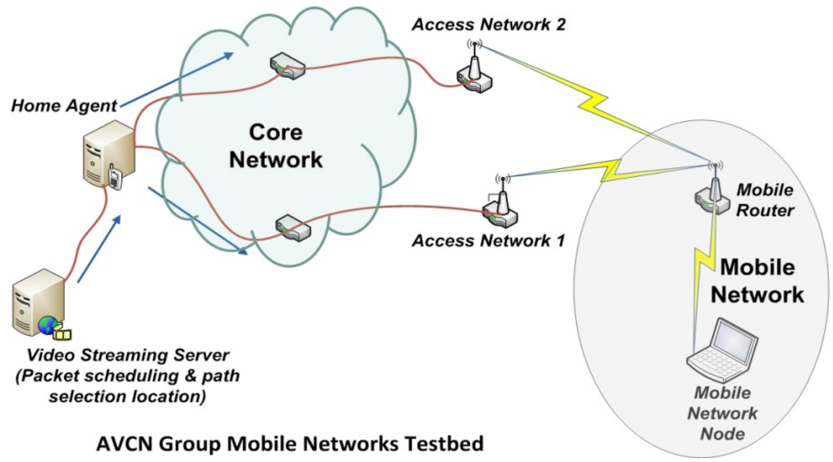

Figure 5. UWS AVCN Mobile Networks Testbed

4.1 Path Control Mechanism

Each of the scheduling algorithms investigated requires knowledge of current network path conditions. We therefore considered the problem of the timely provision of accurate path data in a testbed situation. EDPF assumes a QoS guaranteed available bandwidth for the duration of a streaming session, while LBA considers that the nature of network paths may be time varying both in terms of available bandwidth and delay. We focus on multihomed mobile networks where mobile routers may change their point of attachment and mobile client nodes may join or leave a mobile network during a streaming session. We, therefore, make the reasonable assumptions that the nature of the available network paths can change during a streaming session and that the wired part of any network path will have a significantly higher available bandwidth than the wireless component(s). A number of currently available path monitoring mechanisms were explored as a means of providing current network path information to the scheduling process. However, as they typically required several round trip times to produce an accurate estimation, the metrics being fed to the scheduling process would not provide the current path knowledge required for a real time per packet granularity of decision making.

A mechanism to dynamically change network path conditions and report those changes to the path selection and scheduling module was designed and implemented at each of the core network routers. Each router runs a path control server module which reads path data (available bandwidth and delay) from a configuration file. In this work we used available aggregated bandwidths both above and below the maximum requirement of the streamed content and delays ranging from 20ms to 200ms. Path metrics are changed at a preset interval (configurable in 1 second steps) by signaling the new available bandwidth and/or delay for a path to the Wide Area Network (WAN) emulation module running at a core router's egress (mobile network side) interface. The open source Netem[18] tool is used for this purpose. A successful change of path metrics triggers a path update control message from the core router to the streaming server where the new path data is written to a shared memory location providing real time path information to the scheduling process. A path update acknowledgement is returned to the core router by the streaming server, if this is not received within 250ms the router's path control server retransmits the path data to the streaming server.

Each core router runs its' own autonomous path control server, therefore each path changes independently of the other. Path changes are quickly reported to the streaming server, the typical lag between a path change being made at the core router and the new metrics being available to the scheduler is less than 10ms, thus providing an effective real time feed of path data to the scheduling process. Figure 6 gives a diagrammatic representation of the testbed path update mechanism. Path update control messages have a maximum size of 60 bytes and a maximum frequency of one message per path per second, as such they represent a negligible overhead. The placement of path controls on the egress interface of the core routers permits the network path to be controlled as the streamed media content arrives at the wireless access points which are generally considered to be the bottleneck in paths to mobile networks.

This approach is similar to the virtual choke point[19] proposed by Bianchi et al. In our testbed the one way wireline delays (other than the induced delay) are less than 0.5ms and the available bandwidth (prior to throttling) is 100 Mbits/sec. The mobile networks mobility management application introduces a further one way delay of < 0.5 ms. We consider these delays to be negligible, but add 1ms to our overall path delay calculations to take account of them.

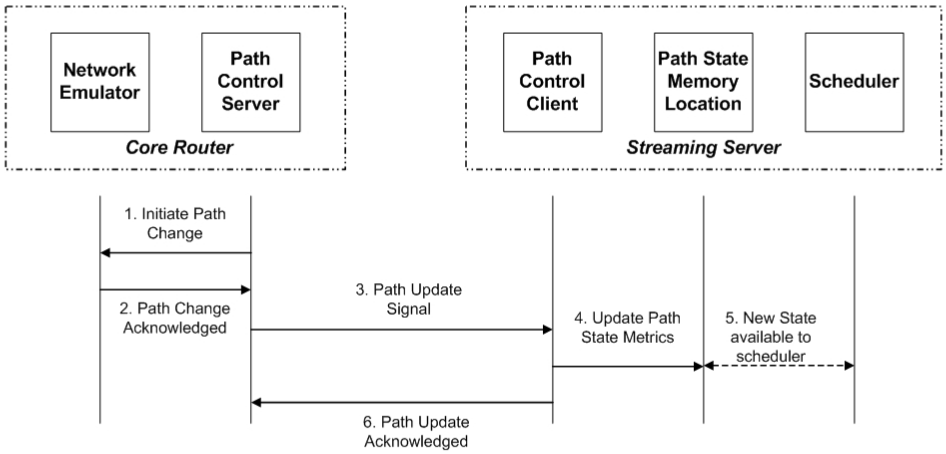

Figure 6. Real Time path Update Mechanism

4.2 Path switching mechanism

We make the reasonable assumption that the point at which network paths from streamer to client diverge in a 'real' network is unlikely to occur at a multi-interfaced streaming server as proposed by Jurca & Frossard, but rather at a router either within the streaming server's local network or in the wider infrastructure. In common with the EDF based proposals, we choose the primary point of divergence of paths to be the mobility agent known as the Home Agent (HA) within the mobile network nodes home network. However, whereas EDF based schemes assume the HA has only one outgoing interface to the mobile node with routing being handled by the infrastructure, we consider a multihomed mobile networks scenario where the HA is a router with distinct network interfaces for each available network path. In LBA, the scheduling algorithm resides on the streaming server, while EDF based schemes place it at the notional point of path divergence (HA). EDPF places the burden of computationally intensive scheduling algorithms on routers which may already be handling large numbers of traffic flows, with the potential that an HA of a busy mobile network might be required to handle many such scheduling processes at the same time. We also considered a possible bi-directional implementation where the roles of server and client could be reversed with scalable media content being streamed from the mobile client. Such a reversal would mean that the resource limited mobile router (MR) would become the point of path divergence. Although Kofler et al[20] demonstrated a resource constrained wireless router capable of successfully supporting the simultaneous adaption of a small number (around 5 or 6) of SVC streams, their adaptation mechanism required less computational resources than any of the scheduling algorithms we considered. The numbers of users in public transport oriented mobile networks could easily be much greater than this demonstrated capability. In order to remove the computational burden from the HA and MR routers, which already handle mobility management, and to limit significant software upgrades required to both streamer and mobile client, we place the scheduling mechanism at the streaming server. By implementing the path switching scheme proposed by Wang et al[7], we were able to provide a solution where the scheduling and path selection process at the streamer could remotely change the path taken by SVC packets. A simple reception module listens on the HA for path switching commands from the scheduling process on the streamer and switches outgoing interfaces for the SVC stream in response to those commands. This requires only a minimal upgrade to routers and makes very light use of router resources in comparison to running a scheduling or SVC adaption process on them. This is in line with our aim that a bi-directional process could be implemented using the MR as the switching point. The path switching mechanism is shown in figure 7.

Our testbed provides a practical multihomed MIPv6 based mobile network which realistically represents a 'real' mobile network and provides timely path characteristics to the scheduling algorithm thus; permitting the real time evaluation of proposed streaming algorithms.

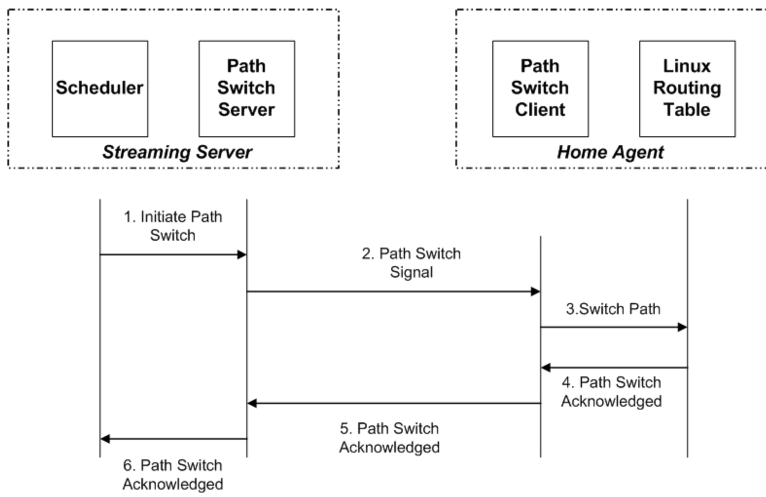

Figure 7. Path Switching Mechanism

5. IMPLEMENTATION

5.1 Streaming Framework

SVEF[11] is an open source testing and evaluation tool for single path IPv4 transmission of stored SVC content. As next generation networks use the IPv6 version of the Internet Protocol, we have rewritten the network interface of SVEF to support both IPv4 and IPv6 address families and the MIPv6 based mobility management software used in mobile networks. AVC compliant Video Coding Layer (VCL) base layer NAL units are encapsulated with the preceding non-VCL type 14 prefix NAL unit, containing scalability information for the base layer packet in a Single Time Aggregation Packet (STAP). All other NAL units, (both VCL and non-VCL) are encapsulated in a single packet. These pseudo- RTP packets are then transported using the connectionless UDP protocol, with packet fragmentation being handled in the Linux kernel by the IP layer.

The authors of LBA and EDPF assumed that packet metrics such as size, relative importance/priority and decoding deadline are available to the scheduling algorithm, but did not specify a mechanism for providing this data. In this work we consider how to obtain this data for both stored and real time streaming scenarios. The SVEF trace file is extended by rewriting the existing preprocessing module to calculate the relative decode deadline (in relation to the first NAL unit in a stream) and priority weighting of each NAL unit. The preprocessor permits the easy sorting of the trace file to test a number of algorithms and streaming scenarios. In our implementation we include the capability of applying priority weighting scheme which reflects the ability to fully utilize the granularity offered by the three dimensional scalability. However, in this paper we; limit our evaluation to the use of the same priority weightings used by the authors of LBA thus providing a fair comparison with other schemes.

Each NAL unit n_i has a decoding deadline t_i^d and an arrival time at the client t_i^a. On order to, remove the need for streamer and receiver synchronisation the decoding deadline of a NAL unit is calculated relative to the decode time $t_{i=1}^d$ of the first NAL unit in the stream. This is assumed to be its arrival time $t_{i=1}^a$ at the client. ($\therefore\ t_{i=1}^d = t_{i=1}^a = 0$) The relative decoding deadline of subsequent NAL units is calculated using the video frame number f_i to which a NAL unit belongs and the sequence frame rate δ in frames per second (fps), such that $t_i^d = \frac{f_i}{\delta} \times 1000$ (milliseconds). A NAL unit can only be useful in the decoding process if it arrives within its own decoding window as determined by its decoding deadline and the playback delay Δ at the client. If $t_i^a \leq (t_i^d + \Delta)$, the NAL unit is useable in the decoding process. The arrival time t_i^a calculation varies for each algorithm and is described in the relevant section of this paper.

We also consider the fact that the default JSVM[21] encoder output order already provides some measure of NAL unit prioritization by outputting frames in a prediction hierarchy order rather than an absolute frame number order. Prediction hierarchies vary according to the GOP structure of the stream.. Figure 8 provides an example.

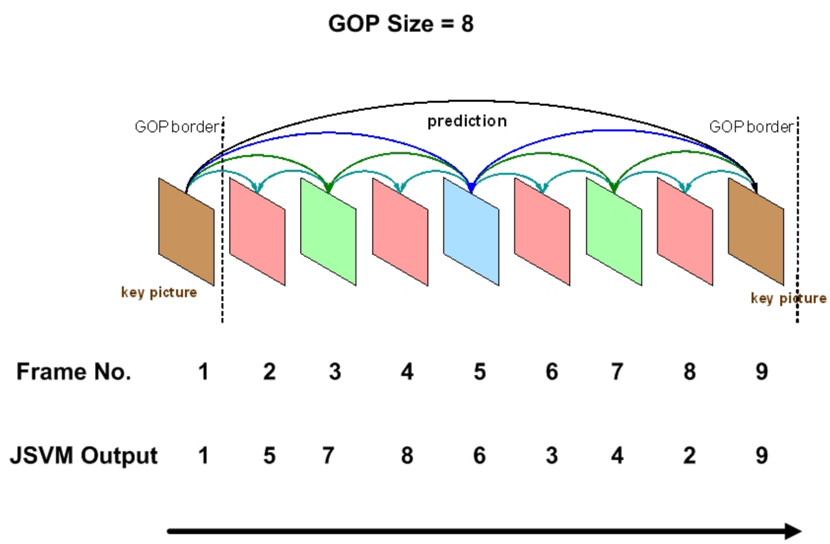

Figure 8. Encoder NAL unit output order

SVEF is further extended by inclusion of three separate path selection and packet scheduling algorithm modules (EDPF, LBA & our optimized algorithm for multihomed mobile networks). New path status reporting and path switching control modules are also added to the framework. The original framework was designed for use with stored streaming content, we extend this to include real time streamed content by introducing 'on the fly' processing based on a configurable pre-fetch window size which is given as a number of GOP's. All packet scheduling and path selection functions are sited at the streaming server. This provides a level playing field for evaluating the relative efficiency of each algorithm. For each implementation we provide a practical interpretation of the proposed algorithm.

5.2 EDPF Implementation

For each packet p_i, EDPF considers the time at which the packet arrives at the scheduling point a_i and the size of the packet L_i. For each path *l,* EDPF takes account of the one-way wire line delayD_l, the bandwidth negotiated by QoS mechanisms on the wireless link B_l and the time at which the wireless interface next becomes free A_i. The arrival time $t_i^a$of packet $p_i$at the client node is given as –

$$t_i^a = MAX(a_i + D_l, A_i) + L_i/B_l \quad (1)$$

In our implementation, we interpret each of these components as above with the exception that $D_l$is the delay introduced by our path control mechanism D_l^p plus the mean wire line delay in our testbed D_l^w such that $D_l = D_l^p + D_l^w$; and B_l is the bandwidth set by our path control mechanism. Each packet is scheduled onto the path offering the earliest arrival time. As each path selection decision is made, the chosen path is compared to the current (last used) path and, if different, the new path number is passed to the path switching module running on the streaming server. A path switching control signal is then sent to the Home Agent, where the path change is initiated and an acknowledgement is returned to the streaming server. EDPF does not consider mobility related tunneling overheads when calculating packet size L_i, nor path switching costs in the overall wireline delay D_l. The trace file used to control the packet stream uses the default JSVM encoder output order.

5.3 LBA Implementation

LBA varies from EDPF in three main aspects. Firstly; the way packet arrival times are calculated includes consideration of intermediate node buffer constraints. Secondly; a packet prioritization scheme is introduced which includes packet sorting at the sender. Thirdly, performance is enhanced by using an explicit ancestor scheduling check.

We use the same packet arrival time calculations as EDPF since manual calculations indicated that the difference between the two approaches in the testbed is negligible. A larger testbed, with more intermediate nodes, may require a full implementation of the LBA path delay calculations. Packets are firstly arranged in descending order of priority. Each packet's ancestors are scheduled, and if any dependencies cannot be met the packet is dropped. Packets are then either scheduled onto the earliest delivery path or dropped if they will not arrive within their decode window. Further modification of the offline SVEF preprocessing tool allows specification of the read ahead window size (as a number of GOP's). A revised trace file with NAL units grouped by the specified number of GOP's and sorted firstly by priority weighting and then by decoding deadline is produced. NAL units now arrive at the scheduler in the order specified in LBA. Using a trace driven model provides a common platform for all schemes and allows for future investigation of stored streaming scenarios. The number of operations performed on the NAL units within the scheduler's input buffer is also reduced. Ancestor checks can only be performed within the scheduler's current window of knowledge. The frame number and H.264 SVC scalability information carried at RTP packet level are used to confirm that all ancestors have been scheduled thus ensuring that a packet, which cannot be decoded due to unmet dependencies, is not transmitted to the client. Each path which can deliver the packet within its decode window is considered to be a candidate path, the packet is sent on the candidate path offering the earliest delivery time. If no candidate path is found the packet is dropped at the scheduler. Path switching is handled in the same way as with EDPF.

5.4 Optimised Algorithm for Multihomed Mobile Networks

Overheads consist of the network overhead (RTP, UDP & IPv6) of an RTP packet which does not require fragmentation θ_f and the mobility tunneling overhead $\theta_m$which will be dependent on the level of nesting in the mobile network. Each additional IP packet of a fragmented NAL unit will also have its own network overhead θ_a and a mobility overhead added. The number of fragments $F_i$of RTP packet r_i is derived by dividing the RTP payload size by the MTU. The full network size L_i^n is calculated from the initial single packet overhead and the extra overheads of each fragment.

$$L_i^n = L_i + (\theta_f + \theta_m) + F_i(\theta_a + \theta_m) \tag{2}$$

We also consider the delay γ_s introduced by the path switching mechanism and compensate for this by adding this delay to the estimated arrival time of each path lo other than that the current path. The arrival time on the current path lc for RTP packet r_i is given as

$$t_i^a = MAX(a_i + D_{lc}, A_i) + L_i^n / B_{lc} \tag{3}$$

The arrival time for RTP packet $r_i$on each alternate path lo is calculated as

$$t_i^a = MAX(a_i + D_{lo}, A_i) + L_i^n / B_{lo} + \gamma_s \tag{4}$$

We next seek to reduce the path switching overhead by means of a judicious trade off between optimum bandwidth aggregation and minimum switching delay. This is achieved by enhancing our LBA implementation. After a packet has been through the dependency check, the arrival time on the current path is calculated using (3). If the packet will arrive at the client within its decode window, it is scheduled onto that path, removing the need for path switching and the associated delay. When the current path is unable to deliver a packet on time, (4) is used to calculate the arrival times on alternate paths and, as with LBA, either find a suitable candidate path or drop the packet at the scheduler. Switching from the current path to a new path may result in packets arriving out of sequence at the client. We, therefore, ensure that the first packet sent on a new path will not arrive before its decode window opens. The decode window for a packet opens when $t_i^a = t_i^d$ and closes when $t_i^a = (t_i^d + \Delta)$. By holding the packet at the scheduler until its estimated arrival time is $t_i^a = t_i^d$, we reduce the probability of out of sequence arrival.

6. PERFORMANCE EVALUATION

In this section, we empirically compare the performance of the three algorithms (EDPF, LBA and OSPSA) in our mobile networks testbed. Publicly available video sequences were encoded using the JSVM reference software and the performance of each algorithm compared in terms of packet delivery statistics and statistical video quality measurements. Versions of the *Soccer* sequence with 4CIF (704x576) resolution were used at frame rates of 30 and 60 fps, together with versions of the *Bus* sequence with QCIF (176x144) resolution and frame rates of 15 and 30 fps. Each was encoded with a base layer and two MGS scalability layers.

In each case, the number of RTP packets dropped by the streamer, the number arriving at the client and the number arriving both on time and in the correct order are measured. The first measurement is achieved by a mechanism incorporated into the scheduler which records dropped packets to a log file. This log file contains a record of packet ID, frame number, priority weighting and reason for dropping for each packet dropped at the scheduler. We employ three categories of reason 1) *A packet on which the current packet depends was not scheduled and thus the current packet cannot be decoded*, 2) *There is no viable path to deliver the packet on time* and 3) *Sending this packet would adversely alter the arrival time of a previously scheduled packet*. The same log file also records the number and frequency of path switching operations and any path delay or bandwidth changes reported to the scheduler by the path control mechanism. It should be noted that the EDPF implementation does not drop packets at the scheduler. The delay introduced by each individual path switching operation is also recorded. The reception module at the client also produces a log file which records every packet received and the time (relative to the first packet in the stream) at which it was received. An enhanced version of the SVEF *nalufilter* tool has been produced which provides a report detailing the number of packets arriving at the client and the number lost during transmission. Every packet which arrived too late to be useful in the decoding process or out of sequence is also identified.

One of the limitations of the current version of the JSVM decoder is its inability to correctly deal with packets which have unmet dependencies. We, therefore use the SVEF *framefiller* mechanism to firstly generate a filtered copy of the video sequence containing only those packets that correctly arrived at the client on time to be of use in the decode process and had all ancestor packet available at the decoder. The simple SVEF frame filler routine is then applied to conceal missing parts of the sequence. This reconstructed video sequence is compared to the original sequence using the JSVM reference software PSNR comparison tool.

6.1 Path Switching Overhead

We firstly investigated the typical path switching delay in our testbed. The streamer log file recorded the time taken for each path switching operation to be acknowledged by the HA and the estimated arrival time at the client (relative to the first packet in the stream). As the arrival time (relative to the first packet in the stream) was also recorded at the client we were able to compare the variance between projected and actual arrival times. From this we established that the average path switching cost was 137ms. Figure 9 shows the delay induced by each path switch for a typical 30fps sequence.

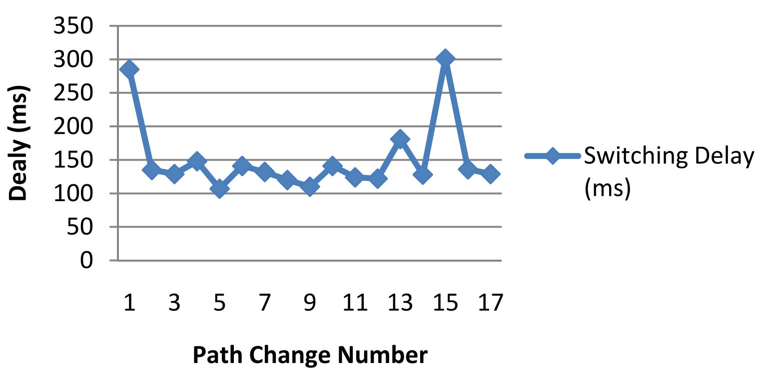

Figure 9. Path switching delay

After establishing the average path switching cost we implemented a further version of LBA in which the calculated arrival time on any path (other than the current path) had the average path switching delay of 137ms added. This provided a path switching compensated version of LBA that could be used for comparison to our optimised algorithm.

6.2 Additional Network Overheads

In order to determine the effect on performance of added mobility overheads, we compared two versions of our optimised algorithm, one which took account of the additional overheads and one which only used the payload size (as per our EDPF & LBA implementations) when calculating expected the arrival time of a packet. We used both the *Soccer* and *Bus* sequences across the full range of frame rates. When the overheads were not considered, the number of packets arriving at the client that arrived after their decode deadline or could not be used because an ancestor had arrived after its decode deadline increased by an average of 6.1% .The *Bu*s sequence, which has fewer large packets, performed better than the *Soccer* sequence. The number of base layer packets failing to arrive on time also increased (from 0.05% to 0.8 % for the S*occer* sequence).Received video quality was reduced for both sequences with the average PSNR for the *Soccer* sequence being reduced by 0.36 dB and the *Bus* sequence by 0.14 dB. It can, therefore, be seen that the effects on video quality of added mobility overheads in mobile networks are non negligible.

6.3 Path Switching Frequency

Given the significant delay attached to each path switching operation, we investigated the path switching frequency of each algorithm. Table 1 shows the average path switch frequencies expressed as a ratio of *Number of Path Switches: Number of Packets Transmitted.* Path switching frequency varied considerably between algorithms and was much higher when paths had similar bandwidth and delay characteristics. It should also be noted that there was some variation on switching frequency across video sequences with the Bus sequence producing a higher path switching frequency than the Soccer sequence. We consider this to be due to the lower variation size of the packets in the Bus sequence which leads to better load balancing in LBA, from which it can be inferred that the nature of video stream and the encoding parameters used in the JSVM encoder (which affect packet size) have an impact on path switching frequency.

Table1. Path Switching Frequency Ratios.

	EDPF	LBA	LBA Compensated	Optimised Algorithm
Equal Paths	1:2.2	1:2.1	1:2.8	1:3.1
High Differential	1:3.9	1:4.1	1:14.3	1:26.1

The effect of increased path switching frequency is a cumulative induced delay which leads to packets arriving later than their decode deadline. With the EDPF and LBA implementations where path switching costs are not considered, the cumulative delay quickly leads to all packets arriving after their decode window has closed. The path switching compensated version of LBA performs better and maintains delivery of enough packets within the decode deadline to provide an acceptable quality of video, however it still has a significantly higher switching ratio than our optimised algorithm. Figure 10 shows the relative performance (in terms of PSNR) of each of the implemented schemes.

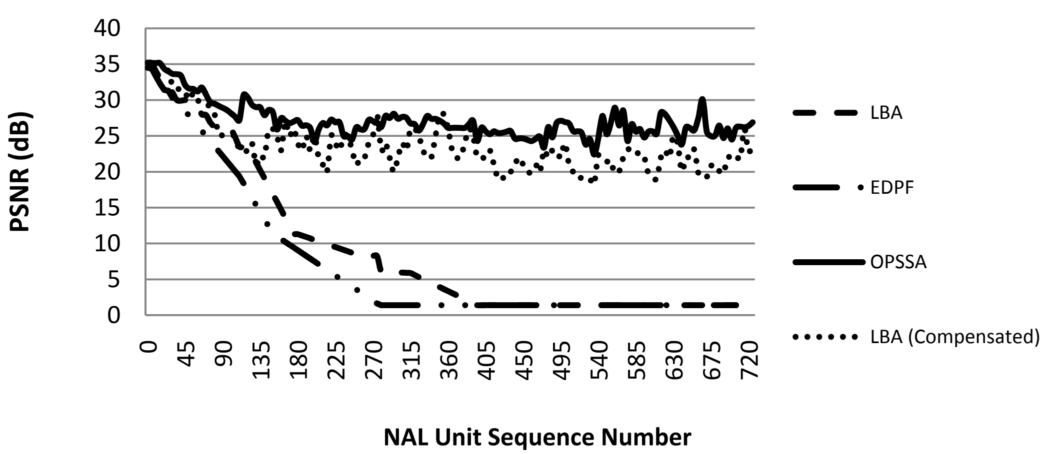

Figure 10. PSNR Comparison

7. CONCLUSIONS

The contributions of this paper are twofold. We provide both a practical testbed for SVC streaming and propose a novel scheduling scheme. By designing and building a realistic multihomed mobile networks testbed we were able to practically implement and evaluate real-time multipath streaming algorithms for H.264 SVC. Our flexible streaming framework allows the testing of path selection and scheduling algorithms under a range of easily configured network conditions, streaming window sizes and packet priority schemes. Due to its modular nature, our framework can be extended to evaluate new algorithms which may be proposed in the future. Empirical evaluation of the three implemented algorithms (EDPF, LBA and our optimised scheme) has highlighted the incompatibility of 'always best connected' requirement and the concept of optimal bandwidth aggregation across all network paths.

We show that the added network overheads associated with mobility, if unconsidered, lead to degradation in the quality of a received video stream and that the algorithms which did not consider path switching costs were unable to deliver an acceptable quality of video in multihomed mobile networks. Our optimised algorithm significantly reduced the number of path switching operations, even when compared to a path switching compensated version of LBA and delivered the highest quality of video across the range of sequences tested. When using equal paths the PSNR performance of path switching compensated LBA was closer to that of our optimised algorithm but was still typically 0.4 to 1 dB lower. However when paths with a high bandwidth differential were used, the difference was substantially higher and ranged from 2.3dB to 5dB. We, therefore show that by sacrificing optimal bandwidth aggregation; which occurs when paths re equal or have only a small differential; in favour of a reduced path switching frequency, that a higher quality of received video can be achieved.

In this work we chose not to fully implement and test the enhanced packet prioritization scheme incorporated in our framework in order to provide a better comparison of the other novel aspects of our design. Future work will focus on the development of this area.

REFERENCES

[1] Schwarz H., Marpe D., and Wiegand T., "Overview of the Scalable Video Coding Extension of the H.264/AVC Standard," IEEE Transactions on Circuits and Systems 17, 1103-1120 (2007).
[2] Wiegand T., Sullivan G., Bjontegaard G., and Luthra A., "Overview of the H.264/AVC video coding standard," IEEE Trans. Circuits Syst. Video Technol. 13(7), 560-576 (2003).
[3] Perkins C., "Mobile IP," IEEE Communications Magazine 35(5), 84-99 (1997).
[4] Johnson D. , Perkins C., and Arkko J., "Mobility support in IPv6," IETF RFC3775, (2004).
[5] Devarapalli V., Wakikawa R., Petrescu A., Thubert P., "Network mobility (NEMO) basic support protocol," IETF RFC 3963, (2005).
[6] Wang Q., Atkinson R., and Dunlop J., "Design and evaluation of flow handoff signalling for multihomed mobile nodes in wireless overlay networks," Computer Networks 52, 1647-1674 (2008).
[7] Wang Q. et al, "QoS-aware network-supported architecture to distribute application flows over multiple network interfaces for B3G users," Wireless Personal Communications 48, 113-140 (2009).
[8] Wenger S., Hannuksela M., Stockhammer T., Westerlund M. and Singer D., "RTP Payload format for H.264 video," IETF RFC 3984, (2005).
[9] Wenger S., Wang Y.-K., Scheirl T., and Eleftheriadis A, "RTP payload format for SVC video," IETF Internet Draft, draft-ietf-avt-rtp-svc-22, (2010).
[10] Zahariadis Th., " ASTRALS Project presentation," IBC Amsterdam 2007, (2007).
[11] Detti A., Bianchi G., Pisa C., Proto F.S., Loreti P., Kellerer W., Thakolsri S., and J. Widmer J., "SVEF: an open-source experimental evaluation framework for H.264 scalable video streaming," Proc. 2009 IEEE Symposium on Computers and Communications, 36-41 (2009).

[12] Kormaz T. and Krunz M., “Routing multimedia traffic with QoS guarantees,” IEEE Trans. Multimedia 5 , 429-443 (2003).
[13] Wang Z. and Crowcroft J., “Quality-of-service routing for supporting multimedia application,” IEEE J. Sel. Areas Commun. 14, 1228-1234, (1996).
[14] Chebrolu K. and Rao R., "Bandwidth aggregation for real-time applications in heterogeneous wireless networks," *IEEE Transactions on Mobile Computing* 5, 388–403 (2006).
[15] K. Chebrolu and R. Rao, "Selective frame discard for interactive video," Proc. 2004 IEEE International Conference on Communications , 4097-4102 (2004).
[16] Jurca D. and Frossard P., "Video packet selection and scheduling for multipath streaming," IEEE Transactions on Multimedia 9, 629–641 (2007).
[17] Fernandez J., Taleb T., Guizani M. and Kato N., “Bandwidth aggregation-aware dynamic QoS negotiation for real-time video streaming in next-generation wireless networks,” IEEE Trans. Multimedia 11(6), 1082-1093 (2009).
[18] http://www.linuxfoundation.org/collaborate/workgroups/networking/netem
[19] Bianchi G., Detti A., Loreti P., Pisa C., Proto F., Kellerer W., Thakolsri S., and Widmer J., "Application-aware H. 264 scalable video coding delivery over wireless LAN: Experimental assessment," Proc. IWCLD'09. Second International Workshop on Cross Layer Design, 1–6 (2009).
[20] Kofler I., Prangl M., Kuschnig R., and Hellwagner H., “An H.264/SVC-based adaptation proxy on a WiFi router,” Proceedings of the 18th International Workshop on Network and Operating Systems Support for Digital Audio and Video - NOSSDAV ’08, 63 (2008)
[21] Reichel J., Schwarz H., and Wien M., “Joint Scalable Video Model 11(JSVM 11),” Joint Video Team, Doc. JVT-X202, (2007).