Transmission Optimization of Digital Compressed Video in Wireless Systems

Pietro Camarda and Domenico Striccoli
Politecnico di Bari
Italy

1. Introduction

Digital multimedia transmission is one of the most significant technological challenges of our time. In the last few years, new innovative services have been developed allowing users to benefit of high quality multimedia contents through a wide variety of transmission standards (e.g., Digital Video Broadcasting, 3rd Generation cellular networks, etc.). The modern video compression techniques, like the MPEG or H.264/AVC standards (Mitchell et al., 1996), allow the final users to experience high quality video decoding with a consistent bandwidth saving. At the same time, the significant progress in wireless communication networks with high Quality of Service (QoS) guarantees has brought to a development of innovative multimedia services, whose flexibility can satisfy the mobility requirements peculiar to the ever growing number of wireless terminal users. This process has been also encouraged by an integration between fixed and mobile networks where final users can combine telecommunications, information technology and entertainment services offered from various operators. The most highlighting and well known example is the Universal Mobile Telecommunication System (UMTS) network, where streaming servers providing video are typically located in wired packet-data or telephone networks. Video data cross the core network and are then broadcasted or multicasted to mobile users through the wireless Radio Access Network. The development of value added services on cellular networks like UMTS are regulated by the 3rd Generation Partnership Project (3GPP), that defines the standard for 3rd generation cellular networks (Holma & Toskala, 2004), dedicating also a special work group for streaming video (3GPP TSG-SA; 3GPP TS 26.234; 3GPP TR 26.937). Another example of wired-wireless integrated network infrastructure is the Digital Video Broadcasting for Handheld terminals (DVB-H) system, where multimedia contents are transmitted through the IP network, exploiting the same DVB Terrestrial (DVB-T) infrastructure (ETSI TR 101 190; ETSI EN 300 744; ETSI TR 102 377), and are received on several kind of terminals like smartphones, Personal Digital Assistants (PDAs), notebooks, etc.
Video transmission over wireless terminals has introduced several new issues, peculiar to these environments, that need to be investigated, like Doppler and multipath noises with consequent signal degradation, handover, etc. These issues become also more critical in the case of video streaming, where frame losses should be minimized to keep a high quality video decoding. Power saving has to be maximized for an efficient multimedia reception

Source: Digital Video, Book edited by: Floriano De Rango,
ISBN 978-953-7619-70-1, pp. 500, February 2010, INTECH, Croatia, downloaded from SCIYO.COM

because of terminals batteries with limited capacity. Another important aspect is that compressed video is generally characterized by a bit rate highly variable in time. This makes the optimal allocation of bandwidth resources very difficult: a static bandwidth assignment with data highly fluctuating in time could easily bring to an underutilized channel, or consistent frame losses.
In this chapter we will face the problem of the dynamic optimization of Variable Bit Rate (VBR) video transmission in wireless environments. Schedule at transmission side is varied during stream running according to the varying system conditions. The goal is to maximize the network channel utilization and reduce lost bits, to perform a continuous video playback on wireless terminals without any quality degradation.
Two different, but structurally similar, environments will be analyzed: the DVB-H system and the UMTS network. Transmission plans at server side will be proposed that take into account not only the VBR data, but also the channel bandwidth and the buffering capacity of mobile terminals. In the UMTS scenario data transmission will be varied by taking into account the user interactivity. The proposed algorithms are compared with the transmission techniques already known by literature and guidelines; their effectiveness will be validated in several simulation scenarios.

2. Video transmission frameworks

2.1 Digital Video Broadcasting for Handheld terminals

In the recent years, several aspects have contributed to the exponential growth of Digital Video Broadcasting and innovative services delivered through wireless networks: an improved digital video quality with consistent bandwidth saving thanks to the MPEG video compression standard; a better allocation of frequency resources, the possibility of user interactivity, and the reception of high-quality services during terminal motion. In this context, DVB for Handheld terminals (DVB-H) is able to carry multimedia services through digital terrestrial broadcasting networks. Since the Terrestrial DVB (DVB-T) infrastructure is able to serve both fixed and mobile terminals, DVB-T and DVB-H systems share almost the same common physical layer of the ISO-OSI protocol stack; nevertheless, DVB-H introduces additional features in the Physical Layer, Data Link Layer and Network Layer of the stack (ISO/IEC 7498-1), as represented in Fig. 1. This system is designed for VHF and UHF frequency bands, in the same spectrum assigned to analogue TV.
At Physical Layer, single channels can occupy 6MHz, 7MHz or 8MHz bandwidths. To multiplex a wide number of signals the Orthogonal Frequency Division Multiplexing (OFDM) is adopted, because of its high efficiency in rejecting the multipath noise typical of wireless links. DVB-T allows two different transmission modes: the 2K and 8K. They refer to the number of OFDM subcarriers multiplexed in a "symbol" for transmission. The 2K mode is suitable for small Single Frequency Networks (SFNs). In the SFNs all transmitters carry the signal in the same frequency range, and in small SFNs there is a reduced distance among transmitters. The 8K mode is adopted for small, medium and large SFN networks. In addition, DVB-H allows also the 4K mode. It can be used both for single transmitter operation and for small and medium SFNs, providing also a higher robustness towards noise and allowing very high speed reception.
Each OFDM subcarrier is modulated with the Quadrature Amplitude Modulation (QAM) or Quaternary Phase Shift Keying (QPSK). The transmission system is hierarchic since two

different MPEG streams are multiplexed in the modulated signal: a High Priority (HP) stream and a Low Priority (LP) stream.
Video data are grouped into fixed-length packets of 188 bytes, including 1 byte of synchronization header. The 187 information bytes are then coded through a proper pseudo-random bit redistribution that minimizes the energy dispersion (the Adaptation Energy Dispersal). In the Outer Coder a Error Correction Code is applied; the adopted code is the Reed-Solomon that adds 16 parity bytes generating packets of 204 bytes. The so obtained data are then interleaved in the Outer Interleaver; a bit coding in the Inner Coder allows a further protection against transmission errors in the aerial channel. A further interleaving performed by the Inner Interleaver allows the correct associations among bits that will then be mapped by the Mapper on the OFDM subcarriers.
The Transmission Parameter Signalling (TPS) information is added to the OFDM symbol. It consists of an adding number of OFDM subcarriers conveying only signalling information about modulation techniques, code rates, transmission modes, etc.; this operation is called Frame Adaptation. (ETSI EN 300 744). The OFDM symbols are then organized in frames and a guard time interval is set between consecutive symbols. The so packetized data are then sent to a D/A converter and then to the terrestrial channel.
At the Data Link Layer, the data broadcast service shall insert data directly in the payload of MPEG-2 Transport Stream packets (ISO/IEC 13818-1). DVB standard specifies four methods for data broadcasting: Data Piping, Data Streaming, Multi-Protocol Encapsulation (MPE) and Data Carousel, together with additional Program Specific Information (PSI) and Service Information (SI) that are proper data descriptors (ETSI EN 301 192). Specifically, MPE is well suited to the delivery of streaming services. MPE supports the delivery of other protocols, giving more flexibility, and is also suitable to implement additional features specific to DVB-H environments, that is, Multi Protocol Encapsulation - Forward Error Correction (MPE-FEC) and Time Slicing.

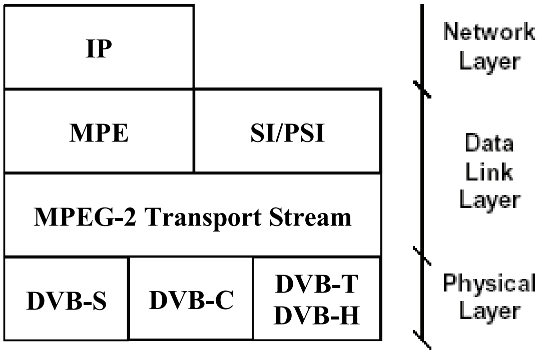

Fig. 1. Protocol stack (layers 1-3) for DVB-H transmission

MPE-FEC is introduced to improve the system robustness towards Doppler noise and impulse interferences. To this aim, an additional level of error correction is introduced at the MPE layer. By calculating parity information and sending it in MPE-FEC sections, error-free datagrams can be received also in very bad reception condition (ETSI EN 302 304).

Time Slicing is instead introduced to improve terminal power saving and manage handover. At network layer, IP protocol is adopted. The DVB-H payload consists of IP datagrams incapsulated into MPE sections and multiplexed together with classical MPEG-2 services (ETSI EN 301 192). An example of a DVB-H system for IP service transmission is illustrated in Fig. 2, where DVB-H services are multiplexed with traditional MPEG-2 services.

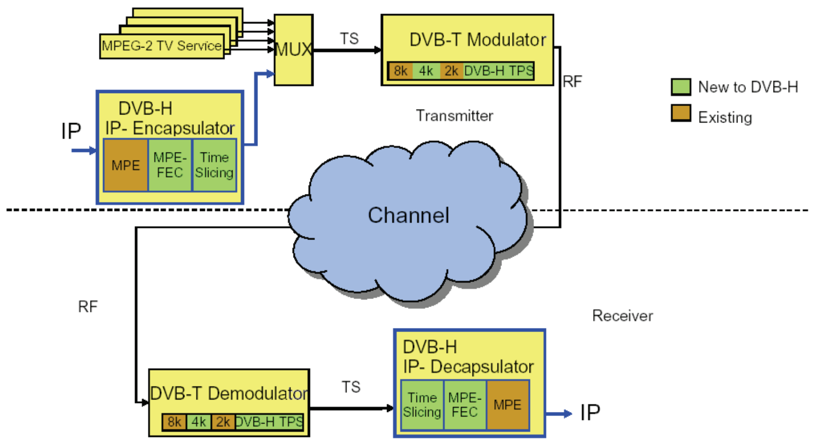

Fig. 2. Conceptual DVB-H system structure

2.1.1 Time slicing

Time Slicing is a transmission technique thought to improve mobile terminal performance in terms of energy saving and handover. Service data are transmitted in “packets” or “bursts” periodically repeating in time with the classical Time Division Multiplexing (TDM) technique. Service Bursts are interspaced by “off-times”, i.e., time intervals between two consecutive bursts of the same service, where no service data are decoded but data of other multiplexed services are transmitted. The energy saving is achieved because receiver remains active only for a fraction of time, during the reception of the chosen service, and switches off during that service off-times. Furthermore, off-times can be exploited to manage handover, that is, cell changes, through proper signal monitoring, without service interruptions. Stream decoding should always be continuous and without losses, so the transmitted bit rate in a burst must be consistently higher than the average bit rate required for the classical DVB-T transmission. Fig. 3 illustrates the comparison between DVB-T and DVB-H transmission of four services.

The main parameters for the single time-sliced service are illustrated in Fig. 4. The Burst Size (BS) represents the total Network Layer bits transmitted in a burst, included the MPE-FEC and Cyclic Redundancy Check code (CRC-32) headers. The Burst Bitrate (BB) is the constant bit rate of the time-sliced stream during the burst transmission. The Constant Bitrate (CB) is the average bit rate required by the stream not time sliced, as would be transmitted in classical DVB-T systems. The Burst Duration (BD) is the time interval from the burst beginning to its end. The Off-time ”(Ot) is instead the time interval between two consecutive bursts.

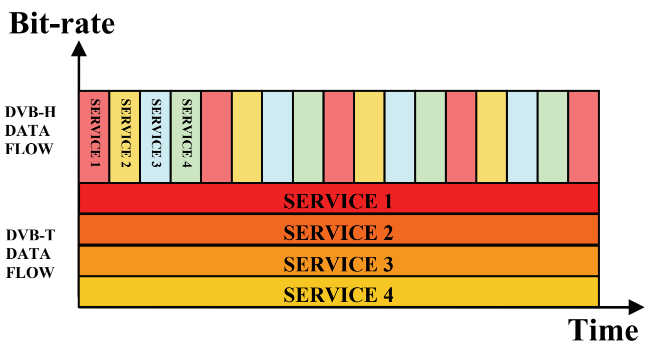

Fig. 3. Comparison between DVB-T and DVB-H transmission

In the sequel, we will define the Burst Cycle (BC) as the time interval between the beginning of two burst of the same service. For each service it holds that $BC = BD + Ot$.
The time slicing implementation allows a consistent power saving at receiving side since the receiver remains inactive during off-times, and the power consumption is reduced. Furthermore, off-times can be exploited to perform handover without any service interruption.
Continuous data decoding is theoretically always guaranteed by receiver data buffering. The burst size is in fact buffered in the client memory during burst durations, and consumed during the subsequent Off-times. Obviously, sufficient buffering is needed at receiving side for continuous and lossless decoding; this condition can be reached by properly regulating the main burst parameters (burst duration and burst bitrate).

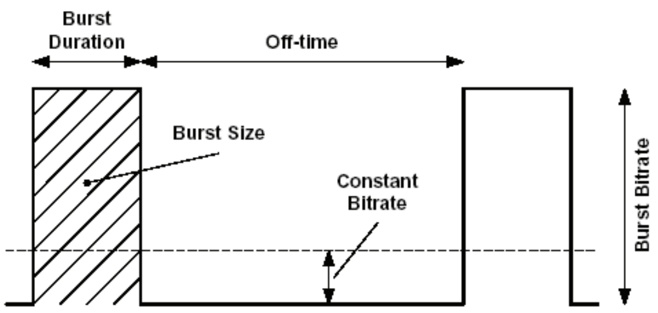

Fig. 4. Time Slicing parameters for the single service

2.2 UMTS networks

Another scenario of great interest in video transmission context is the stream delivery over 3rd generation wireless networks. The 3G networks bring some important enhancements to the previous 2G cellular networks, improving at the same time the already existing services. Specifically, it should:

- Support broadband services, both point-to-multipoint and point-to-point;
- Allow new low-cost terminals of small weight and size, simple to use for the users;
- Support different QoS levels for the wide variety of available services;

- Support an efficient network resource assignment;
- Provide a more flexible rating, depending from the session duration, the transferred data and the desired QoS;
- Implement a wider set of bit rates, depending from the specific service and mobility issues.

The UMTS network is thought as a group of logical networks, each one accomplishing specific functions, as illustrated in Fig. 5 (Uskela 2003).

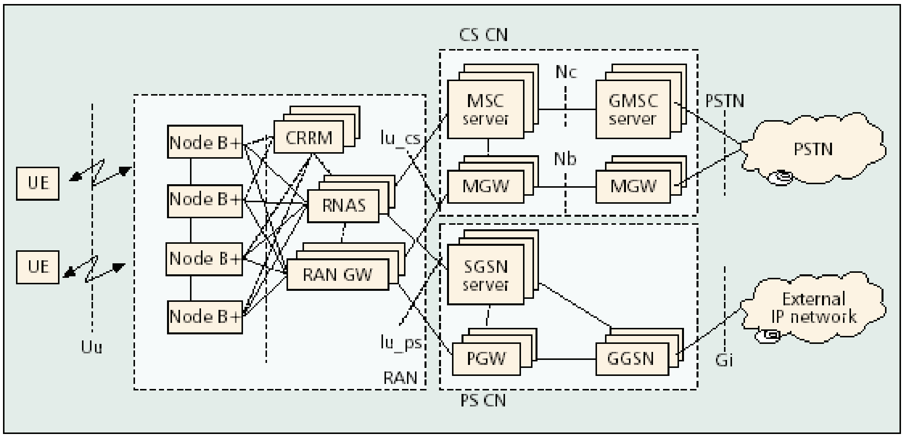

Fig. 5. The 3G network architecture

The Core Network (CN) is the part of the UMTS network that provides services to final users; it can be connected to different types of networks supporting different communication protocols. The CN is composed by the Circuit Switched CN (CS CN) and a Packet Switched CN (PS CN). Regarding the CS CN, the Mobile services Switching Centre (MSC) switches the CS transactions and manages the user mobility. It is interconnected to external networks (like for example the PSTN networks in Fig. 5) through the Gateway MSC (GMSC). The MSC is subdivided into MSC server and Media GateWay (MGW), where all user data are managed. A MSC server can manage more MGWs, allowing a higher network scalability when new implemented services increase data flow. In fact, whenever this happens, it is sufficient to increment the MGWs number.

The PS CN is composed by two elements: the Serving GPRS Support Node (SGSN), that manages the different sessions (including various QoS requirements) and mobility, and the Gateway GPRS Support Node (GGSN), that performs, among the other things, the QoS mapping among different networks, the packet filtering and the connection with other packet switching external networks (such as Internet). The streaming server providing multimedia services can reside either in the external packet data network, or just before the GGSN.

The PS CN is connected to the Radio Access Network (RAN) through the RAN GateWays (RAN GW) and the RAN Access Server (RNAS). The RAN GW is a routing point for the user traffic between CN and UTRAN. All radio resource management functionality is pooled in the Common Radio Resource Manager (CRRM). It is thus possible to provide the radio resource management network-wide to all radio technologies, which allows an efficient utilization optimization. The RNAS main task is to provide an interface to the CN.

RAN GWs and RNAS form the Radio Network Controller (RNC). Each SGSN can manage hundreds of RNCs.
All processing related to the radio interface of the RNC is relocated into the Node B+, whose main functions are, among others: macro diversity control, handover related functions, power control and frame scheduling. Data are provided to the User Equipment (UE), which is the radio terminal utilized by the user to exploit the UMTS network services.

2.2.1 Feedback information through RTCP

To guarantee a good quality multimedia streaming while minimizing losses at client side, there is a need for scheduling at transmission side. As previously explained, when the network or the client conditions vary during a session, bad effect can appear at client side due to the buffer overflows and underflows events. A very good way for scheduling optimization at transmission side is that the streaming server has knowledge of the "status" of the network and the client terminal. This information is performed through the Real Time Control Protocol (RTCP) (3GPP TS 26.234), that carries control information between the media player on the user equipment and the streaming server. RTCP packets carry the main statistics on the media stream, that can be exploited by the streaming server to regulate the transmission rate whenever network congestion and/or losses at receiving side occur. RTCP main role is to provide a feedback information on the quality of the stream delivery.
Streaming content is instead carried through the Real-time Transfer Protocol (RTP) (Schulzrinne et al., 2003), and the streaming sessions are administered by the Real Time Streaming Protocol (RTSP), an application protocol that allows external actions (pause, record, rewind, etc.) on stream data.
RTCP can be a solution for the wireless multimedia streaming over IP networks, since it could help improving the quality of the delivered data, that depends on the variable conditions of the wireless links and the user equipments limited amount of memory. To this aim, the 3GPP Packet-switched Streaming Service (3GPP PSS) specifications give a guideline for transmission rate optimization through the RTCP feedback (3GPP TS 26.234). RTCP packets are usually sent with a periodicity of 5 seconds.
The streaming standard introduces the following RTCP parameters:

- *HTSN (Highest Trasmitted Sequence Number)*, that is the sequence number of the last packet sent by the server;
- *HRSN (Highest Received Sequence Number)*, that is the sequence number of the last packet arrived at the client buffer;
- *NSN (Next Sequence Number)*, that is the sequence number of the next packet to be decoded by the client.

The 3GPP TS 26.234 standard considers also the RTCP packet evolution, called the Next Application Data Unit (NADU) packet, whose structure is illustrated in Fig. 6.
The NADU packet fields are:

- *SSRC*, that is the source identifier of the stream this packet belongs to;
- *PD (Playout Delay)*, representing the time interval between two consecutive reports (in milliseconds);
- *NSN (Next Sequence Number)*, the sequence number of the next packet to be decoded;
- *NUN (Next Unit Number)*, the next video frame to be decoded;
- *FBS (Free Buffer Space)*, representing the free residual buffer space at receiving side (in multiple of 64 bytes);
- *Reserved*; these bits are not used.

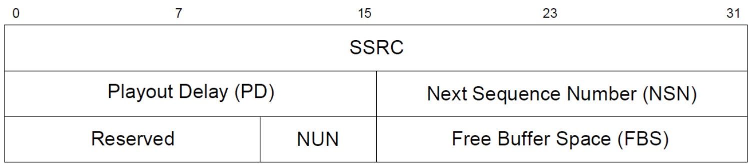

Fig. 6. A NADU packet

Through a NADU packet a high number of information can be derived, such as the number of packets that reached the client, the number of packets stored in the client buffer and decoded by the client. Fig. 7 clearly illustrates these parameters.

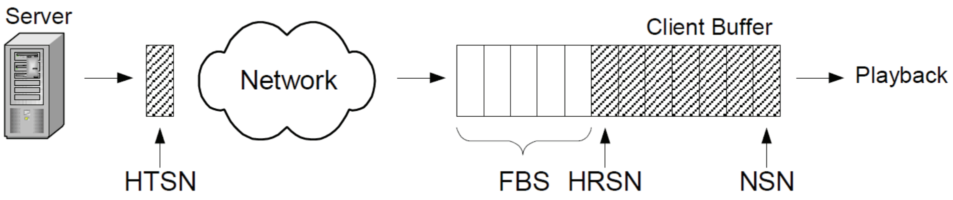

Fig. 7. Feedback information parameters

Packets with a sequence number less than NSN have already been decoded. If the streaming server knows the size of each transmitted packet, the HRSN and NSN parameters provide the occupancy level of the client buffer. It can be noted that this parameter can also be derived by the FBS information if the client sent the buffer size information to server during the connection setup.

3. Principles of video compression

The need of compressing digital video has been felt since 1988, when the standardization effort in this direction was initiated by the Moving Picture Experts Group (MPEG), a group formed under the auspices of the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC). A sequence of pictures with accompanying sound track can occupy a vast amount of storage space when represented in digital form, and also needs a huge amount of channel bandwidth to be transmitted without any quality degradation. This amount of storage and bandwidth can be greatly reduced through specific video compression techniques. In this section we only give a brief overview of the MPEG (further distinguished in the MPEG-1, MPEG-2 and MPEG-4) and H.264/AVC compression techniques, actually the most widely adopted standards.

MPEG video compression is applied to temporally consecutive images (the video frames) of a multimedia stream, exploiting the concept of the similarities between temporally adjacent frames. In fact , except for the special case of a scene change, these pictures tend to be quite similar from one to the next. Intuitively, a compression system should take advantage of this similarity. To this aim, each video sequence is divided into one or more Groups of Pictures (GoP), each one composed of one or more pictures of three different types: I-pictures (Intra-coded pictures), P-pictures (Predictive-coded pictures) and B-pictures (Bidirectionally predictive-coded pictures). Specifically, I-pictures are coded independently, entirely without reference to other pictures. P-pictures obtain predictions from temporally preceding

I-pictures or P-pictures in the sequence, whereas B-pictures obtain predictions from the nearest preceding and/or upcoming I- pictures or P-pictures in the sequence. An example of GoP is represented in Fig. 8. On average, P-pictures have a double size if compared with B-pictures, and size 1/3 if compared with I-pictures (Mitchell et al., 1996). Let us note that MPEG sometimes uses information from future pictures in the sequence; so the order in which compressed pictures are found in the bitstream, called "coding order", is not the same as the "display order", the order in which pictures are presented to a viewer.

The compression efficiency is quantified by the compression ratio, defined as the ratio between the original uncompressed image and the compressed one. The compression ratio depends on the compression level and the frame complexity: a more detailed image will have a lower compression ratio and a higher number of bits, than a less detailed image, both coded with the same compression level.

Each frame is subdivided in groups of samples of luminance (to describe grayscale intensity) and chrominance (to describe colors), called *macroblocks* that are then grouped together in *slices*. Each macroblock is transformed into a weighted sum of spatial frequencies through the Discrete Cosine Transform (DCT) to enhance the compression degree. Further compression is then reached through Motion Compensation (MC) and Motion Estimation techniques. Motion Compensation is introduced because if there is motion in the sequence, a better prediction is often obtained by coding differences relative to areas that are shifted with respect to the area being coded. Motion Estimation is simply the process of determining the motion vectors in the encoder, to describe direction and the amount of motion of the macroblocks (Mitchell et al., 1996).

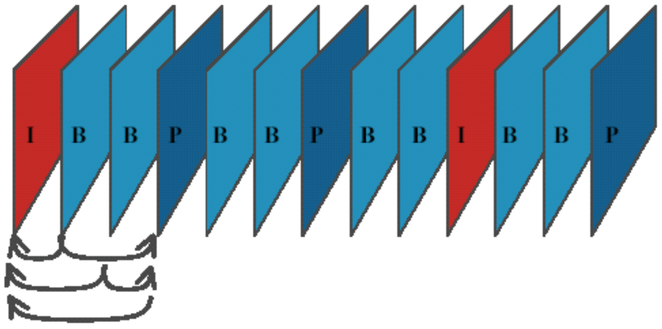

Fig. 8. A Group of Pictures in display order.

Globally, all MPEG standards are based on these concepts. More specifically, MPEG-1 is designed for mean bit rates of 1.5 Mbps, with a video quality comparable to VHS and a frame rate of 25 frames per second (fps) for PAL and 30 fps for NTSC.

MPEG-2 is an extension of MPEG-1. It has been thought for interlaced video coding and utilized for transmission of Standard Definition (SD) and High Definition (HD) TV signals over satellite, cable and terrestrial broadcasting, and the storage of high quality SD video signals onto DVDs (Puri et al., 2004). If compared with MPEG-1, MPEG-2 has a lower compression ratio and a higher mean bit rate.

MPEG-4 was intended for new multimedia applications and services such as interactive TV, internet video etc. (ISO/IEC 14496-2, 2000). It is a living standard, with new parts added continuously as and when technology exists to address evolving applications. The significant advances in core video standard were achieved on the capability of coding video objects, while at the same time, improving coding efficiency at the expense of a modest increase in complexity.

The H.264/AVC standard (ISO/IEC JTC 1, 2003) is the new state of the art of video coding. It promises significantly higher compression than earlier standards. It is also called "MPEG-4 Advanced Video Coding (AVC)". Since the standard is the result of collaborative effort of the VCEG and MPEG standards committees, it is informally referred to as Joint Video Team (JVT) standard as well. The standard achieves clearly higher compression efficiency, up to a factor of two over the MPEG-2 video standard (Horowitz et al., 2003). The increase in compression efficiency comes at the cost of substantial increase in complexity. Moreover, H.264/AVC includes a high number of application-dependent profiles, ranging from low bit rate to very high bit rate applications. The resulting complexity depends on the profile implemented. Like MPEG-2, a H.264/AVC picture is partitioned into fixed-size macroblocks and split into slices. Each slice can be coded by using I, P, B, SP and SI frames. The first three are very similar to those in previous standards, even if the concept of B slices is generalized in H.264 when compared with prior video coding standards (Flierl & Girod, 2003). SP slices aim at efficient switching between different versions of the same video sequence whereas SI slices aim at random access and error recovery.

All the samples of a macroblock are either spatially or temporally predicted. There are also many new features and possibilities in H.264 for prediction, with several types of intra coding supported (Wiegand, 2002). H.264 standard is also more flexible in the selection of Motion Compensation block sizes and shapes than any previous standard (ISO/IEC JTC 1, 2003). The accuracy of Motion Compensation is in units of one quarter of the distance between luminance samples. Prediction values are obtained by applying specific filters horizontally and vertically in the image. The motion vector components are then differentially coded using either median or directional prediction from neighboring blocks.

The H.264 supports also multi-picture motion-compensated prediction (Flierl et al., 2001), in which more than one previously coded picture can be used as reference for Motion Compensation prediction. This new feature requires both encoder and decoder to store the reference pictures used for inter prediction in a multi-picture buffer. Multiple reference pictures not only contribute to the improvement of the compression efficiency, but also to error recovery.

4. Scheduling video transmission

From Section 3 we know that MPEG or H.264 compressed video is characterized by a high bit rate variability due to I, P or B frames, and the scene complexity in terms of details and the amount of motion. Frames are transmitted with a constant frame rate (25 fps for PAL and 30 fps for NTSC). For this reason, these kind of videos are called Variable Bit Rate (VBR) videos. The VBR video transmission introduces a consistent complexity degree for resource assignment, to keep a high QoS degree. A lossless transmission should require a bandwidth assigned to the single video flow equal to its peak rate. Nevertheless, this assignment surely would bring to a bandwidth waste for almost all the video duration, because of the high video bit rate variability (Zhang et al., 1997).

To reduce the high bit rate fluctuation typical of VBR streams, video smoothing techniques have been introduced for video transmission from a server to a client through a network, through the system model illustrated in Fig. 9. Smoothing algorithms exploit client buffering capabilities to determine a "smooth" rate transmission schedule, while ensuring that the client buffer neither overflows nor underflows, and achieving significant reduction in rate variability.

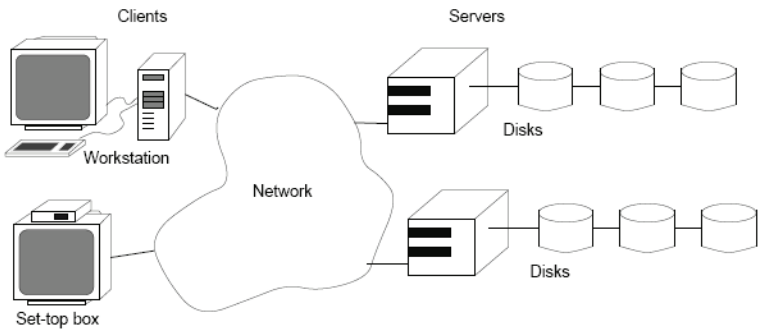

Fig. 9. The server-client model for video delivery.

The basic principle of video smoothing is the "work-ahead", that is, the transmission of video frames ahead of their playback time. As described in (Salehi et al., 1998) a VBR video stream is composed by N video frames, each of them of size f_i bytes $(1 \le i \le N)$. At server side, data are scheduled according to the specific algorithm. At client side, smoothed video data enter the buffer and the original unsmoothed video frame sequence leaves it for decoding and playout. Let us now consider the client buffer model in the k^{th} discrete frame time. A frame time is defined as the time to decode a frame (1/25 s for PAL and 1/30 s for NTSC), and is assumed as the basic time unit. Two curves are built:

$$D(k) = \sum_{i=1}^{k} f_i \tag{1}$$

$$B(k) = b + \sum_{i=1}^{k} f_i = D(k) + b \tag{2}$$

representing, respectively, the cumulative data consumed by the client in k, and the maximum amount of data that can be received by the client in k without overflowing the buffer. It derives that $D(k)$ is the underflow curve, whereas $B(k)$ is the overflow curve. They are both non decreasing curves. A feasible transmission plan $S(k)$ must verify the condition:

$$D(k) \le S(k) = \sum_{i=1}^{k} s_i \le B(k) \tag{3}$$

to guarantee a lossless transmission. $s(i)$ represents the smoothed stream bit rate in the i^{th} frame time. The smoothed stream transmission plan will result in a number of CBR segments, and the cumulative smoothed transmission plan is given by a monotonically increasing and piecewise linear path $S(k)$ that lies between $D(k)$ and $B(k)$ (Salehi et al., 1998).

Different types of smoothing algorithms have been implemented (Feng & Rexford, 1999). They are applied to stored video traffic, where all source video data are all available at server side and can be optimally scheduled "off-line". Nevertheless, there is a growing number of VBR live interactive video applications requiring "on-line" smoothing algorithms, to reduce bit rate variability on-the-fly during stream transmission. Online smoothing is effective to reduce peak rate and rate variability in temporal windows of limited size, at the same time performing a on-the-fly computation of smoothing transmission plans. A further optimization of smoothed transmission plan can be obtained by smoothing videos over consecutive windows partially overlapped in time (Rexford et al., 2000). The impact of available bandwidth on scheduling has also been developed in several works (Bewi et al., 2002; Le Boudec & Thiran, 2004; Lai et al., 2005; Lee, 2006). Bandwidth dependent smoothing considers the additional available bandwidth information for improving the efficiency of the stream schedule.

The basic smoothing principles previously mentioned are the starting point for the implementation of the novel transmission techniques proposed in this chapter. Two scenarios will be analyzed in detail: the VBR video transmission in DVB-H systems and the impact of user interactivity on VBR video delivering in UMTS terminals. For DVB-H systems, a online schedule for all the TDM services is generated at server side to prevent buffer overflows and underflows. It is calculated on consecutive temporal observation windows partially overlapped in time and takes into account available bandwidth resources, burst and receiving buffer sizes. The same smoothing principles can be also used to implement an on-line scheduling that takes into account the user interactivity, that can consistently change the status of receiving terminal, in terms of decoded frames and buffer occupancy level. The on-line schedule, performed over partially overlapping temporal windows, reschedules data according to the feedback RTCP information on the terminal status.

4.1 Optimal scheduling of multiservice VBR video transmission in DVB-H systems

As stated in Section 2.1.1, service data are stored in bursts at transmission side and buffered in the client memory during burst times for continuous playing also during the subsequent off-times. We conventionally say that a burst "covers" the off-time if the burst size guarantees the continuous playback on the DVB-H terminal. Obviously, sufficient buffering is needed at receiving side to cover off-times. In fact the burst size must always be less than the memory available in the receiver and the burst bitrate and burst duration can be set accordingly. Let us note that it is quite easy to statically set the burst parameters for CBR streams; it is instead more difficult for VBR videos, e.g. coded with MPEG or H-264/AVC. In these cases, since the video bit rate is highly variable in time, data stored in the client buffer can heavily vary during video reproduction. Statically setting the burst size for the whole video transmission could easily bring to losses at receiving side, because of the insufficient amount of buffered data and/or because of the relatively small receiving buffer size.

This critical aspect in transmission scheduling is further complicated by the available bandwidth information. Service data in fact could be more effectively scheduled if available bandwidth reserved for the single service is a known parameter. A reduced available bandwidth reduces the maximum amount of data filling a burst because it limits the burst bitrate, under the same burst duration. So when the bandwidth assigned to the service is relatively small, burst bitrate is limited and data stored into bursts could not cover the off-time. The proposed Variable Burst Time (VBT) algorithm, an on-line scheduling algorithm suitable for transmission of time-sliced high quality VBR services, tries to overcome these problems. Transmission optimization is performed by dynamically and simultaneously varying the burst durations of the whole set of services transmitted in a temporal window of fixed size, sliding in time. The optimization method takes into account available bandwidth resources, burst parameters and receiving buffer sizes. The goal is the loss minimization of the whole service set. To test its effectiveness, VBT is compared with the classical DVB-H transmission as recommended by the standard (ETSI TR 102 377), that statically sets all service burst durations.

The VBT schedule exploits the same basic smoothing concepts illustrated in Section 4. Its transmission plan aims to prevent buffer overflows and underflows, the only two conditions supposed to bring to bit losses at receiving side. In the specific DVB-H scenario, a service buffer underflow occurs if the burst size is not enough to cover off-time. A buffer overflow occurs instead if the free buffer level is too small to store the burst size.

4.1.1 The VBT implementation

To fully understand VBT, let us first consider the single service schedule, assuming a discrete time evolution with the basic time unit of a frame time (1/25 s for PAL). VBT schedules as many data as possible in each service burst in advance respect to its playback time avoiding buffer overflows and underflows. To perform this step, supposed the receiving buffer size of b bits and f_i the i^{th} frame size (in bits), the two overflow and underflow curves are built according to (1) and (2):

$$F_{under}(k) = \sum_{i=0}^{k} f_i \; ; \; F_{over}(k) = b + \sum_{i=0}^{k} f_i \qquad (4)$$

The resulting schedule $S(k) = \sum_{i=0}^{k} s_i$ is represented in Fig. 10 in a generic burst cycle, where it is supposed that the service burst duration starts in T_{bi} and ends in T_{bs}, and that the service off-time starts in T_{bs} and ends in T_{cycle}. The burst duration is $T_{on} = T_{bs} - T_{bi}$ and the Off-time $T_{off} = T_{cycle} - T_{bs}$. Consequently the scheduled bitrate s_i will be $s_i = Burst\ Bitrate$ in $\left[T_{bi}, T_{bs}\right]$, during data transmission, and $s_i = 0$ in $\left[T_{bs}, T_{cycle}\right]$. The cumulative schedule $S(k)$ will thus increase only in $\left[T_{bi}, T_{bs}\right]$, with slope equal to burst bitrate, and will remain constant in $\left[T_{bs}, T_{cycle}\right]$. Furthermore, let us assume $q_b = S(T_{bi}) - F_{under}(T_{bi})$ as the buffer fill level in T_{bi}. This is the amount of data stored in the client buffer during the previous burst duration and not yet consumed by client at the end of the previous burst cycle. Similarly, $q_e = S(T_{cycle}) - F_{under}(T_{cycle})$ is the buffer fill level at the burst cycle end.

The schedule will be feasible without losses if and only if $F_{under}(k) \le S(k) \le F_{over}(k) \quad \forall k$. If there are some k so that $S(k) > F_{over}(k)$ a buffer overflow occurs in that k; if instead $S(k) < F_{under}(k)$ there will be a buffer underflow. Both these critical conditions are supposed to generate losses at receiving side. Nevertheless, a buffer overflow can easily be avoided by properly regulating the burst bitrate at transmission side, where the receiving buffer size is supposed to be known, so that $S(k)$ cannot never cross $F_{over}(k)$ in $[T_{bi}, T_{bs}]$. A buffer underflow occurs instead because the burst size is relatively small and cannot cover the off-time, both because available bandwidth limits the burst bitrate or because the off-time is relatively long if compared with the burst duration even without any bandwidth limitation. The complication is that a service off-time strongly depends on the other service burst durations; so *all* the burst durations must be simultaneously adjusted in a burst cycle to minimize losses.

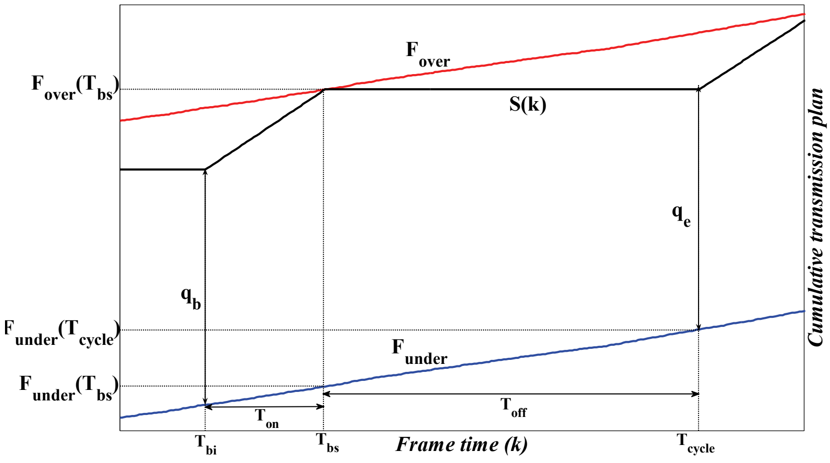

Fig. 10. Single service cumulative transmission plan in a burst cycle

Since VBT is an on-line algorithm, loss minimization is performed on a Temporal Observation Window (TOW) whose length is chosen as a integer number of burst cycles. As explained in (Rexford et al., 2000) a more efficient scheduling can be obtained if the TOWs are partially overlapped in time. The higher the overlapping degree, the better the on-line schedule, even if this comes at a cost of a computational overhead. For a more efficient schedule optimization, VBT also considers partially overlapping TOWs sliding in time. More precisely, we suppose that there are W_S burst cycles in a TOW, and that there are N_S services in a burst cycle. Loss optimization is performed in the considered TOW, that then slides by N_B burst cycles, repeating the optimization procedure in the new TOW. The first N_B burst cycles in the previous step are transmitted and N_B new burst cycles are introduced in the optimization process of the following step. In Fig. 11 this sliding procedure is illustrated, together with the main TOW parameters.

The parameters W_S and N_B influence VBT performance. The TOW length in fact introduces also an initial delay in video playback, since the server must know all service frames to be stored in bursts before calculating the optimal burst durations. On the other side, a larger TOW allows VBT to span a higher number of burst duration configurations, increasing the probability to find a lower minimum for losses.

Also the slide length N_B influences VBT performance. A smaller N_B allows to more efficiently optimize the service burst durations as new service data are scheduled in bursts. In fact, the service data to be scheduled in the last N_B new burst cycles included in the

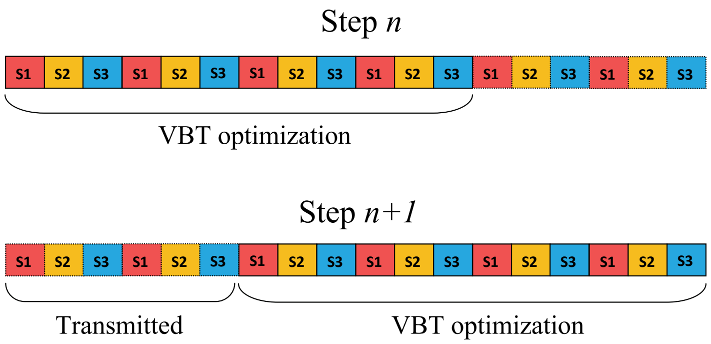
Step *n*
S1 S2 S3 S1 S2 S3 S1 S2 S3 S1 S2 S3 S1 S2 S3 S1 S2 S3
VBT optimization
Step *n+1*
S1 S2 S3 S1 S2 S3 S1 S2 S3 S1 S2 S3 S1 S2 S3 S1 S2 S3
Transmitted
VBT optimization

Fig. 11. The TOW sliding procedure with $W_S = 4$, $N_S = 3$ and $N_B = 2$

$(n+1)^{th}$ step of Fig. 11 refine also the calculation of the service burst durations of the previous $W_S - N_B$ burst cycles already calculated in the n^{th} step, even if the computational overhead increases. The VBT computational complexity also increases with the number of services N_S.
Supposed a generic TOW, we define as *configuration* the n-uple of burst durations (where $n = N_s \cdot W_s$):

$$\overline{T}_{on} = \left(T_{on}^{(1,1)},..,T_{on}^{(N_S,1)}..,T_{on}^{(1,W_S)},...,T_{on}^{(N_S,W_S)}\right) \tag{5}$$

where each $T_{on}^{(i,j)}$ is the i^{th} service burst duration in the j^{th} burst cycle. $T_{on}^{(i,j)}$ is a positive integer multiple of the frame time unit. To find the optimal configuration $\overline{T}_{on,opt} = \left(T_{on,opt}^{(1,1)},...,T_{on,opt}^{(N_S,W_S)}\right)$ that minimizes all service losses a *Total Loss Function* (TLF) is introduced, that considers all the service losses for buffer underflow in the TOW. The first step is to calculate the i^{th} service losses by considering the buffer state $q_e^{(i,j)}$ at the end of the j^{th} burst cycle (see Fig. 10). Let us suppose $T_{on}^{(i,j)}$ the i^{th} service burst duration in the j^{th} burst cycle. We can say that cumulative data filling the buffer (namely, the burst size) are:

$$D_{in}^{(i,j)}(T_{on}^{(i,j)}, W_{av}^{(i,j)}) = BB^{(i,j)} \cdot T_{on}^{(i,j)} \tag{6}$$

Where $BB^{(i,j)}$ is the i^{th} service burst bitrate in $T_{on}^{(i,j)}$. $BB^{(i,j)}$ must be properly calculated to avoid buffer overflow and to be less than the i^{th} service available bandwidth $W_{av}^{(i,j)}$, a constant value assigned to the specific service. That is:

$$BB^{(i,j)} = \min\left\{W_{av}^{(i,j)}, \frac{F_{over}(T_{bs}^{(i,j)}) - \left(F_{under}(T_{bi}^{(i,j)}) + q_b^{(i,j)}\right)}{T_{on}^{(i,j)}}\right\} \tag{7}$$

where $q_b^{(i,j)}$ is the buffer fill level in $T_{bi}^{(i,j)}$.
Cumulative data leaving the buffer at the end of the burst cycle are instead:

$$D_{out}^{(i,j)}(T_{on}^{(i,j)},T_{off}^{(i,j)}) = F_{under}(T_{cycle}^{(i,j)}) - F_{under}(T_{bi}^{(i,j)}) \tag{8}$$

as clearly visible in Fig. 10.
The buffer fill level in $T_{cycle}^{(i,j)}$ is thus:

$$q_e^{(i,j)}(T_{on}^{(i,j)},T_{off}^{(i,j)},W_{av}^{(i,j)}) = q_b^{(i,j)}(T_{bi}^{(i,j)}) + D_{in}^{(i,j)}(T_{on}^{(i,j)},W_{av}^{(i,j)}) - F_{under}(T_{cycle}^{(i,j)}) \tag{9}$$

Losses will occur if and only if the result of (9) is negative, that is, S crosses F_{under} in $T_{cycle}^{(i,j)}$, that is:

$$\begin{gathered} L^{(i,j)}(T_{on}^{(i,j)},T_{off}^{(i,j)},W_{av}^{(i,j)}) = \max\left\{-q_e^{(i,j)}(T_{on}^{(i,j)},T_{off}^{(i,j)},W_{av}^{(i,j)}),0\right\} \\ 1 \le i \le N_S,\ 1 \le j \le W_S - 1 \end{gathered} \tag{10}$$

Let us note that the N_S burst durations of the last burst cycle in a TOW are only used to evaluate the N_S streams off-times for losses evaluation in the $(W_S-1)^{th}$ burst cycle.
$L^{(i,j)}(T_{on}^{(i,j)},T_{off}^{(i,j)},W_{av}^{(i,j)})$ can be derived only after the available bandwidth $W_{av}^{(i,j)}$ and the $\overline{T}_{on}$ vector as defined in (5) have been set, since the generic $T_{off}^{(i,j)}$ of the i^{th} service off-time in the j^{th} burst cycle is given by:

$$\begin{cases} T_{off}^{(i,j)} = \sum_{k=2}^{N_S} T_{on}^{(k,j)} \text{ if } i=1 \\ T_{off}^{(i,j)} = \sum_{k=i+1}^{N_S} T_{on}^{(k,j)} + \sum_{k=1}^{i-1} T_{on}^{(k,j+1)} \text{ if } i>1 \end{cases}, 1 \le j \le W_S - 1 \tag{11}$$

and depends on the burst durations of the other services.
To guarantee fairness among services, the TLF normalizes each service losses to the amount of service data transmitted in the TOW; otherwise, services with lower mean bit rates would be penalized. So the i^{th} service losses are evaluated as:

$$L^{(i)}(\overline{T}_{on},\overline{W}_{av}) = \left(\sum_{j=1}^{W_S-1} L^{(i,j)} \Big/ \sum_{j=1}^{W_S-1} D_{in}^{(i,j)} \right),\ 1 \le i \le N_S \tag{12}$$

where $D_{in}^{(i,j)}$ is given by (6), $\overline{T}_{on}$ by (5) and $\overline{W}_{av} = \left(W_{av}^{(1,1)},..,W_{av}^{(N_S,1)}..,W_{av}^{(1,W_S)},...,W_{av}^{(N_S,W_S)}\right)$ is the available bandwidth vector of the N_S services in a TOW.
Finally, to obtain the TLF the normalized losses calculated in (12) are averaged over the services:

$$M(\overline{T}_{on},\overline{W}_{av}) = \sum_{i=1}^{N_S} L_{fac}^{(i)} \Big/ N_S \tag{13}$$

providing the total amount of losses to be minimized.
The VBT optimal solution must verify the condition:

$$TLF(\overline{T}_{on,opt},\overline{W}_{av}) = \min\left\{TLF(\overline{T}_{on,},\overline{W}_{av}),\overline{T}_{on} \in \mathbb{N}_+^n\right\} \tag{14}$$

where $\mathbb{N}_+^n$ simply indicates the subset of $\mathbb{N}^n$ of all strictly positive natural numbers. The solution to (14) can be found iteratively by numerical methods that find the minimum of a nonlinear bounded multivariable function.

4.1.2 Numerical results

To test its effectiveness, VBT has been compared with the stream transmission as recommended by the DVB-H standard (ETSI TR 102 377) that considers constant burst durations and burst cycles. We call this implementation "Constant Burst Time (CBT) algorithm" for simplicity of notation. Comparison has been performed by multiplexing $N_S = 4$ video streams and reproducing different simulation scenarios. The four chosen video streams, all of length 5.000 video frames, have different quality coding degrees; their main statistics have been listed in Table 1.

Video streams	**Jurassic Park**	**Video Clip**	**Star Wars IV**	**The Silence of the Lambs**
Compression ratio (YUV:MP4)	9.92	38.17	27.62	43.43
Mean frame size (bytes)	3.8e+03	1e+03	1.4e+03	8.8e+02
Min frame size (bytes)	72	31	26	28
Max frame size (bytes)	16745	9025	9370	11915
Mean bit rate (bit/s)	7.7e+05	2e+05	2.8e+05	1.8e+05
Peak bit rate (bit/s)	2.4e+06	8.5e+05	1.2e+06	1.8e+06
Peak/Mean of bit rate	3.15	4.29	4.29	10.07

Table 1. Main video streams statistics

The first proposed experiment shows the influence of the TOW length in VBT losses calculation for three different N_B. The TOW length has been varied from $W_S = 4$ to $W_S = 8$ burst cycles, keeping a constant available bandwidth of 3 Mbps and a receiving buffer size of 320 kB for all services. Fig. 12 depicts the total losses experimented for the services presented in Table 1. As expected, losses decrease with TOW increase since loss optimization for each service is performed over a larger number of burst durations. Furthermore, losses decrease with slide length decrease, because a smaller slide length allows a better refinement in burst durations calculation among subsequent steps. Let us also note that for $W_S = 4$ and $N_B = 3$ losses are evaluated over non overlapped and uncorrelated TOWs; they are thus proportionally much higher than the other experimented cases, since the burst durations refinement through TOW overlapping is not possible.

Total CBT experimented losses are much higher than VBT ones, so they have not been reported in the figure to improve its readability. They have been calculated as follows. The TOW length has been set to the whole streams length to eliminate its influence in CBT loss calculation. The same available bandwidth of 3 Mbps has been considered for all services. The burst duration, the same for all services, has been increased from 3 to 100 frame times (with step 1 frame time) and the minimum of losses calculated in each step has been observed, resulting in 103.2 Mbits. This minimum has been found for a service burst duration of 17 frame times and a burst cycle of 68 frame times. Let us note that CBT losses found in the best scheduling conditions are approximately an order of magnitude higher than the maximum amount of VBT losses (observed for $W_S = 4$ and $N_B = 3$).

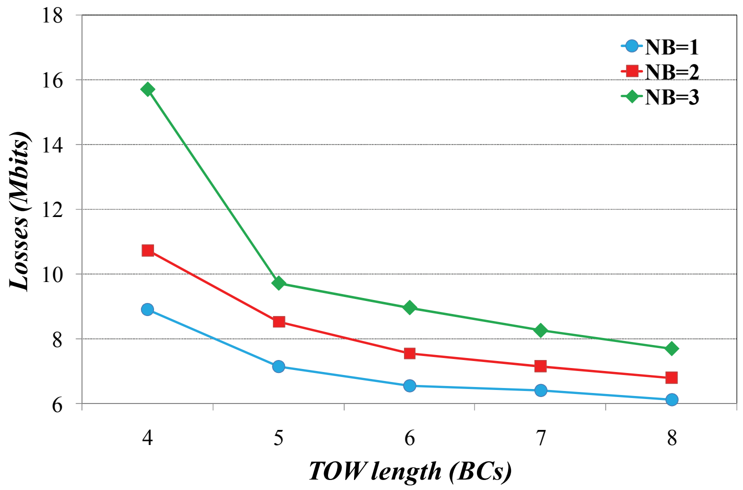

Fig. 12. VBT losses vs TOW length for three different N_B values.

The second proposed experiment illustrated in Fig. 13 shows the impact of the available bandwidth over losses for CBT and VBT. The same four pieces of video streams have been scheduled for transmission in a TOW of $W_S = 8$ burst cycles, using a constant available bandwidth ranging from 1 to 5 Mbps. The receiving buffer size adopted is 320 kB. Numerical results have been displayed for two different slide lengths ($N_B = 1$ and $N_B = 7$). CBT losses have been evaluated as the minimum among all service burst durations ranging from 3 to 100 frame times, as previously explained.
As expected, losses increase with available bandwidth decrease and VBT losses are smaller than CBT ones in all experimented scenarios. Differences between CBT and VBT, and between the two VBT slide lengths, are almost imperceptible for 1 Mbps of available bandwidth, because the dynamic variation of burst durations and the overlapping degree have an almost null effect in reducing losses, that are instead almost exclusively due to the very stringent bandwidth limitation. For increasing available bandwidth, VBT burst duration adjustment is much more effective if compared with the static CBT burst duration assignment.
The last experiment illustrated in Fig. 14 shows the impact of the receiving buffer size over losses for CBT and VBT schedules. Regarding VBT, the TOW length has been set to 8 burst cycles with a slide length of $N_B = 1$ burst cycle. Regarding CBT, the whole services have been scheduled a time with the same procedure illustrated in Fig. 12 for loss minimization. The client buffer sizes range from 128 to 1024 kB, with step of 128 kB. VBT losses have been evaluated for two different available bandwidth values (3 and 4 Mbps), while the available bandwidth for CBT has been set to 5 Mbps to exclude the influence of the bandwidth limitation in loss calculation.

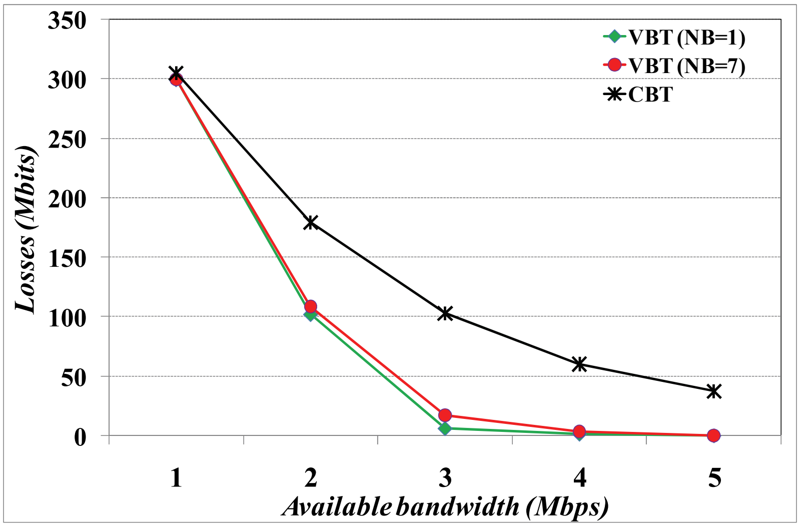

Fig. 13. Losses vs available bandwidth for VBT and CBT schedules.

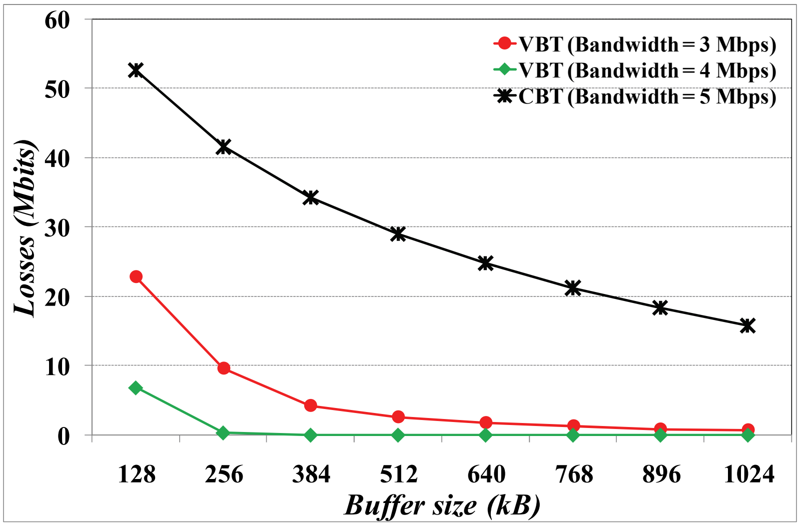

Fig. 14. Losses vs receiving buffer size for VBT and CBT schedules.

As expected, losses decrease with buffer increase for both VBT and CBT. This happens because a larger buffer size allows to store more data at receiving side, reducing the loss probability for buffer underflow. Again VBT losses are always smaller than CBT ones; furthermore, CBT losses are still present even if the available bandwidth is relatively high: with a buffer size of 1024 kB, CBT total losses are still of about 16 Mbits. VBT experimented losses, with a 4 Mbps bandwidth, are instead null right from a buffer size of 384 kB up, testifying that video scheduling through dynamic burst duration adjustment, together with proper buffering at receiving side, can drastically abate losses.

4.2 Scheduling VBR video streams in 3G networks

The main factors that influence the high quality delivery of audio and video contents over UMTS terminals are the highly fluctuating conditions of wireless links and the limited amount of buffering on mobile terminals. UMTS systems should guarantee lossless data delivery despite of the highly variable bandwidth conditions of wireless channel and the high fluctuating data bit rates. Furthermore, a certain degree of user interactivity, that strongly influences data buffering and decoding, should be taken into account to avoid continuous rebufferings and bad media quality, respectively due to receiving buffer underflows and overflows. These problems can be faced through dynamic scheduling at transmission side. In this work we focus on the effects of the user interactivity that modifies the status of the mobile terminal, ignoring the influence of other aspects like the fluctuating channel bandwidth.

In this section we present an on-line scheduling algorithm, the Dynamic-Buffer Dependent Smoothing Algorithm (D-BDSA) suitable for interactive multimedia applications in 3G wireless networks. Scheduling is performed over partially overlapped temporal windows, sliding in time according to the feedback information carried to the streaming server by RTCP packets. Rescheduling is performed because of external user actions (pause, or fast forward) that change the client buffer fill level. The goal is the frame losses reduction.

Let us suppose a streaming server providing VBR video to a 3G terminal. The server reduces the video bit rate variability through the work-ahead MVBA smoothing as developed in (Rexford et al., 2000) calculated over partially overlapped TOWs of N frames, sliding by α frames each step. α is the so called *slide length*. In each step, the first α frames scheduled in the previous step are sent to the client, while α new frames are scheduled together with the remaining $N-\alpha$ frames already scheduled in the previous step. In the on-line smoothing algorithms with a statically assigned α analyzed in (Rexford et al., 2000), it has been experimented that $\alpha = N/2$ is a good compromise between an optimized schedule and a reduced computational overhead. D-BDSA exploits the feedback information provided by RTCP packets to dynamically adjust the slide length according to the user actions. In this analysis, we suppose that there is enough available bandwidth in the UMTS network, so that delays in data transmission and control information across the network can be assumed almost null. For this reason the FBS information completely describes the client status (see Fig. 7). In fact in the generic k^{th} frame time it holds:

$$\begin{aligned} HTSN(k) &= HRSN(k) \\ NSN(k) &= HTSN(k) + FBS(k) - b + 1 \end{aligned} \tag{15}$$

where b is the client buffer size.

Let us now suppose that a NADU packet containing the FBS information arrives to the server in the k^{th} frame time. Let us call this information $FBS_C(k)$ for simplicity, to indicate that it is sent by the client. The server compares the value of $FBS_C(k)$ with the free expected buffer level derived by the schedule, that we call $FBS_S(k)$, that is simply given by:

$$FBS_S(k) = B(k) - S(k) > 0 \tag{16}$$

With $B(k)$ and $S(k)$ given by (2) and (3) respectively. The server knows this information because it calculates the transmission plan for each k. Let us point out that the lower bound

$D(k)$ defined in (1) is the cumulative amount of bits consumed by the client during video playback, so the server calculates the transmission plan always assuming continuous playback at client side. So, if the user performs a video playback, it surely will be $FBS_S(k) = FBS_C(k)$. If instead the user performs other actions like for example fast forward, or pause, it will be $FBS_S(k) \neq FBS_C(k)$ depending on the specific user action. For example, if the user pauses the stream, $FBS_C(k) < FBS_S(k)$ because data only enter the client buffer, while the server supposes that frames also leave the buffer for playback. And vice versa for the fast forward action. In this case:

- Video data must be rescheduled by the server according with the new updated $FBS_C(k)$ information coming from the client, to prevent frame losses;
- The frequency of the feedback information must be increased as a function of the $|\Delta B| = |FBS_S(k) - FBS_C(k)|$ displacement, to more quickly adjust the schedule according to the real client buffer status.

The two critical conditions experimented at client side are a buffer underflow and a buffer overflow. Nevertheless, whenever the client buffer underflows because of a fast forward action, when the server performs a rescheduling in k it knows exactly the number of the last decoded packet $HTSN(k)$, since the buffer is empty, so it can send scheduled frames starting by $HTSN(k)$. The user will thus experiment only rebufferings with consequent delays in frame decoding; frame losses for buffer underflow will *never* occur. In the case of a buffer overflow instead losses surely will occur because the buffer is full and the server continuously sends data. This implies that $HTSN(k) > HRSN(k)$, and the server will not be able to send lost frames again. Both these critical conditions can be prevented as more as α is small, because the server can more quickly react to a buffer variation through rescheduling. This suggests the implementation of the relationship $\alpha(\Delta B)$. Remembering that:

$$|\Delta B| = |FBS_S(k) - FBS_C(k)| \Rightarrow 0 \leq |\Delta B| \leq b \tag{17}$$

by (17) it is clear that the higher $|\Delta B|$ the more likely a user action that changes the free client buffer level compared with the one calculated by the server. Rescheduling should thus be as more frequent as $|\Delta B|$ increases. The proposed solution is hyperbolic relationship between α and $|\Delta B|$:

$$|\Delta B| = \frac{a_1}{\alpha} + a_2 \tag{18}$$

This kind of relationship has been chosen because small $|\Delta B|$ increments bring to high α decrements, which is typical of a hyperbolic relationship. a_1 and a_2 can easily be derived by imposing the two bound conditions:

$$\begin{cases} |\Delta B| = 0 \Rightarrow \alpha = N/2 \\ |\Delta B| = b \Rightarrow \alpha = 1 \end{cases} \tag{19}$$

The maximum slide length has been chosen $\alpha = N/2$ since, as previously mentioned, it is the best trade-off between the optimality of the schedule and the algorithm computational overhead. Imposing the (19) and solving the system:

$$\begin{cases} a_1 + a_2 = b \\ \dfrac{2a_1}{N} + a_2 = 0 \end{cases} \Rightarrow \begin{cases} a_1 = \dfrac{Nb}{N-2} \\ a_2 = -\dfrac{2b}{N-2} \end{cases} \tag{20}$$

it derives:

$$\alpha = \frac{Nb}{(N-2)|\Delta B| + 2b} \tag{21}$$

The (21) is exploited by the server to dynamically update the frequency of rescheduling.
D-BDSA is summarized as follows:

1. When a RTCP packet comes to the server in k, the server compares $FBS_C(k)$ with $FBS_S(k)$ and calculates $\Delta B = FBS_S(k) - FBS_C(k)$;
2. The server calculates the next scheduled transmission time $k_1(\Delta B) = k + \alpha$ of the RTCP packet through (21), and sends this information back to the client through specific RTCP packets (Schulzrinne et al., 2003), supposed to be immediately available to client;
3. The server updates the $NSN(k)$ information exploiting the second of (15) calculated in k;
4. The server reschedules video data in a TOW of N frames, considering $NSN(k)$ calculated in the step 2 as the first frame to be decoded. Then sends the first α scheduled frames, until a new $FBS_C(k_1)$ information arrives from the client.

4.2.2 D-BDSA performance

In this Section, we test D-BDSA effectiveness by comparing it with the same BDSA scheduling algorithm, but with a constant slide length α. We call this version of BDSA Static-BDSA (S-BDSA), setting $\alpha = N/2$. All time units are expressed in frame times. Comparison between S-BDSA and D-BDSA has been made by simulating the transmission of 70.000 video frames of the "Jurassic Park" video, MPEG-4 coded with high quality, with a sequence of the simulated user external actions summarized in Table 2.

The first proposed experiment is illustrated in Fig. 15. It shows the influence of the TOW length N on losses for D-BDSA and S-BDSA. N is varied from 500 frame times (20 seconds) to 1.000 frame times (40 seconds) with step 50 frame times (2 seconds). The client buffer size has been chosen of 1 Mbyte. D-BDSA losses are always smaller than S-BDSA ones, thanks to the D-BDSA dynamic change of the slide length. For both schedules, losses decrease with N decrease because a smaller N means a smaller α, both for S-BDSA ($\alpha = N/2$) and even more for D-BDSA (where $1 \le \alpha \le N/2$) and a resulting increased feedback frequency that reduces loss probability. On the other side, let us note that a smaller N increases the algorithm computational complexity since a higher number of TOWs is needed to schedule the whole video stream.

The second proposed experiment illustrated in Fig. 16 shows the S-BDSA and D-BDSA performance for different client buffer sizes. Losses have been calculated by choosing the same piece of video stream used in the previous simulation and the same sequence of user actions listed in Table 2, with *N*=600 frame times. The buffer size has been increased from 64 kbytes until 2 Mbytes, with increasing powers of 2.

Action Number	Action type	Duration (frame times)	Starting time (frame time)	Ending time (frame time)
1	Playback	5.000	1	5.000
2	Pause	2.000	5.001	10.000
3	Fast Forward 4x	500	10.001	10.500
4	Playback	2.000	10.501	12.500
5	Fast Forward 2x	2.000	12.501	14.500
6	Playback	2.000	14.501	16.500
7	Fast Forward 4x	700	16.501	17.200
8	Playback	2.000	17.201	19.200
9	Pause	2.000	19.201	21.200
10	Playback	2.000	21.201	23.200
11	Fast Forward 2x	1.000	23.201	24.200
12	Playback	10.000	24.201	34.200
13	Pause	5.000	34.201	39.200
14	Playback	Until stream end	39.201	End of Stream

Table 2. Sequence of the user actions on the client terminal

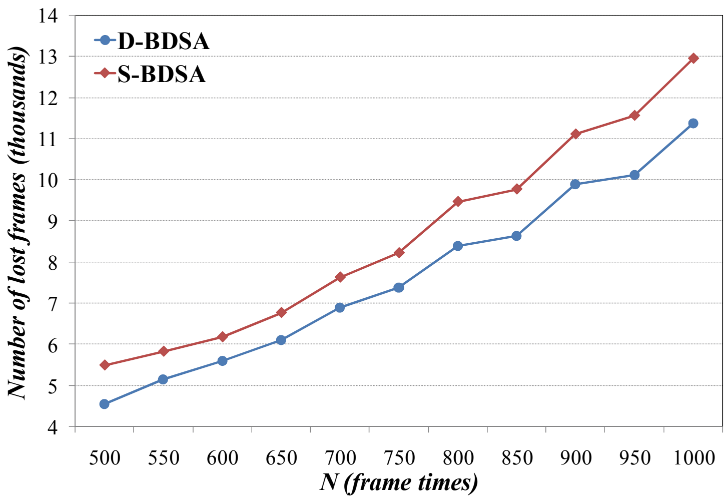

Fig. 15. Lost frames vs TOW length for S-BDSA and D-BDSA

As shown by Fig. 16, losses decrease with buffer increase since larger client buffers reduce the buffer overflow probability. D-BDSA losses are always smaller than S-BDSA ones. For smaller buffer sizes (64, 128 and 256 kB) losses are high in both cases, even if they decrease more quickly for D-BDSA. This happens because the video flow is compressed with high quality, with a relatively high number of bits per frame that cannot be stored by the client buffer, on average increasing the buffer overflow probability. The same result has been experimented also for other values of N and/or other stream types. For a 4 Mbyte buffer and higher, losses are null in both cases.

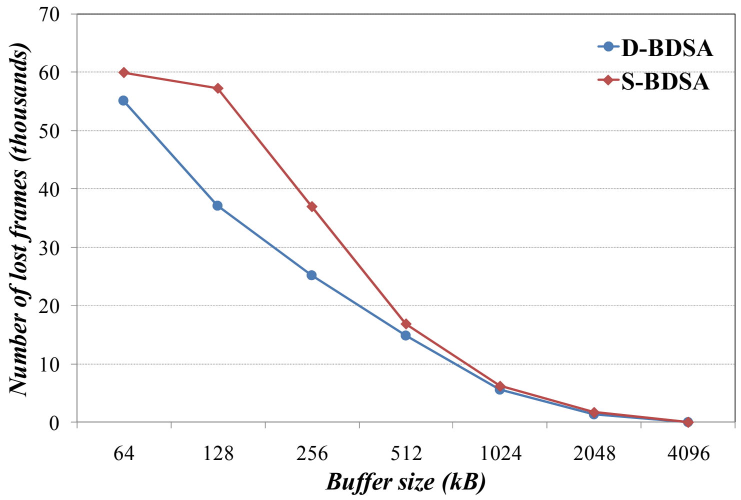

Fig. 16. D-BDSA and S-BDSA losses vs client buffer size.

5. Conclusions and future work

Studies illustrated in this chapter show that dynamic scheduling of VBR streams in wireless systems is very effective in reducing losses. Two different transmission scenarios have been analyzed where dynamic scheduling can be fruitfully applied: the DVB-H system, where a number of time multiplexed services share the same channel resources, and the UMTS network where schedule calculated by the streaming server is influenced by user actions reported back to server by RTCP packets.

Regarding DVB-H, the simultaneous dynamic variation of all service burst durations is of great help in reducing losses when VBR videos are transmitted. It is performed by taking into account service data, receiving buffer size and available bandwidth. Further work in this direction can be done in the improvement of the optimization method that finds the minimum of a nonlinear function of several variables. The method implemented in the proposed study finds local minimum, that in any case provides very good results. Nevertheless the optimization method could be further refined by finding a global TLF minimum, possibly with a relatively lower computational cost.

Regarding the UMTS network, the dynamic schedule derives from user actions that modify the client buffer status. Simulation have been performed over different types of video streams, TOW lengths and buffer sizes, testifying the effectiveness of the proposed method. Performance results are strongly influenced by the sequence of the user actions and especially by the slide length values calculated by the server. Further improvements in this direction can be done by testing other methods for dynamic α calculation and adopting different scheduling algorithms than MVBA, that can more quickly react to the varying terminal conditions to further reduce frame losses. The real status of the UMTS core network, together with its buffers, could also be modeled to analyze the network behavior towards losses.

6. References

3GPP Technical Specification Group Services and System Aspects (TSGS-SA). TR 26.937. *Transparent end-to-end Packet Switched Streaming Service (PSS). RTP usage model (Release 6)*. Version 6.0.0, March 2004.

3GPP Technical Specification Group Services and System Aspects (TSGS-SA). TS 26.234. *Transparent end-to-end Packet Switched Streaming Service (PSS). Protocols and codecs (Release 6)*. Version 6.3.0, March 2005.

Bewi, C., Pereira, R., Merabti, M. (2002). *Network Constrained Smoothing: Enhanced Multiplexing of MPEG-4 Video*. Proc. 7th International Symposium on Computers and Communications (ISCC'02), pp. 114-119, July 2002.

ETSI EN 300 744. *Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for digital terrestrial television*.

ETSI EN 301 192. *Digital Video Broadcasting (DVB); DVB specification for data broadcasting.*

ETSI EN 302 304. *Digital Video Broadcasting (DVB); Transmission System for Handheld Terminals (DVB-H).*

ETSI TR 101 190. *Digital Video Broadcasting (DVB); Implementation guidelines for DVB terrestrial services; Transmission aspects*.

ETSI TR 102 377. Digital *Video Broadcasting (DVB); DVB-H Implementation Guidelines.*

Feng, W.-C., Rexford, J. (1999). *Performance Evaluation of Smoothing Algorithms for Transmitting Prerecorded Variable-Bit-Rate Video*. IEEE Transactions on Multimedia, Vol. 1, No.3, , pp. 302-313, September 1999.

Flierl, M., Wiegand, T., Girod, B. (2001). *Multihypothesis Pictures for H.26L*. IEEE ICIP 2001, Greece, 2001.

Holma, H. & Toskala, A. (2004). *WCDMA for UMTS : Radio Access for Third Generation Mobile Communications*. John Wiley & Sons, 3rd edition, ISBN 978-0471720515, New York.

Horowitz, M., Joch, A., Kossentini, F., Hallapuro, A. (2003). *H.264/AVC Baseline Profile Decoder Complexity Analysis*. IEEE Transactions on Circuits and Systems for Video Technology, Vol. 13, no. 7, pp. 704-716, 2003.

ISO/IEC 13818-1. *Information Technology - Generic Coding of Moving Pictures and Associated Audio Information - Part 1: Systems*.

ISO/IEC 14496-2. (2000). International Standard: 1999/Amd1:2000, 2000.

ISO/IEC 7498-1. *Information Technology – Open System Interconnection – Basic Reference Model: The Basic Model*.

ISO/IEC JTC 1.(2003). *Advanced video coding. ISO/IEC FDIS 14496-10*. International Standard, 2003.

Lai, H., Lee, J.Y., Chen, L. (2005). *A Monotonic Decreasing Rate Scheduler for Variable-Bit-Rate Video Streaming*. IEEE Transactions on Circuits and Systems for Video Technology, vol.15, n.2, pp.221-231, February 2005.

Le Boudec, J.-Y., Thiran, P. (2004). *Network Calculus: A Theory of Deterministic Queueing Systems for the Internet*. Book Springer Verlag, May 2004.

Lee, M. (2006). *Video Traffic Prediction Based on Source Information and Preventive Channel Rate Decision for RCBR*. IEEE Transactions on Broadcasting, vol.52, n.2, pp.173-183, June 2006.

Mitchell, J.L., Pennebaker, W.B., Fogg, C.E. & LeGall, D.J. (1996). *MPEG video compression standard*. Chapman & Hall, ISBN 978-0412087714, London.

Puri, A., Chen, X., Luthra, A. (2004). *Video Coding Using the H.264/MPEG-4 AVC Compression Standard*. Elsevier Science, Signal Processing: Image Communication, Sept. 2004.

Rexford, J., Sen, S., Dey, J., Kurose, J., Towsley, D. (2000). *Online Smoothing of Variable-Bit-Rate Streaming Video*. IEEE Transactions on Multimedia, vol.2, n.1, pp.37-48, March 2000.

Salehi, J.D., Zhang, Z.-L., Kurose, J., D. Towsley. (1998). *Supporting Stored Video: Reducing Rate Variability and End-to-End Resource Requirements Through Optimal Smoothing*. IEEE/ACM Transactions On Networking, Vol.6, N.4, pp. 397-410, August 1998.

Schulzrinne, H., Casner, S., Frederick, R., Jacobson, V. (2003). *RTP: A transport protocol for real time application*. RFC 3550, July 2003.

Uskela, S. (2003). *Key Concepts for Evolution Toward Beyond 3G Networks*. IEEE Wireless Communications, pp.43-48, February 2003.

Wiegand, T. (2002). *Joint Final Committee Draft*. Doc. JVT-E146d37ncm, Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG (ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6), November 2002.

Zhang, Z.L., Kurose, J., Salehi, J.D., Towsley, D. (1997). *Smoothing, statistical multiplexing, and call admission control forstored video*. IEEE Journal on Selected Areas in Communications, Vol.15, no.6, pp. 1148-1166, August 1997.