Streaming for Vehicular Users via Elastic Proxy Buffer Management

Vincenzo Mancuso, Università di Palermo
Giuseppe Bianchi, Università di Roma Tor Vergata

ABSTRACT

In this article we refer to the market of vehicular networks, where groups of customers located in the same public vehicle (e.g., a train or bus) connect to a terrestrial network through a wireless/satellite backbone link. Elastic buffering is a proxy management technique devised to decouple the multimedia information retrieval rate on the network backbone from the playout streaming rate at the user terminal. It has been shown in the past that the application of elastic buffering mechanisms in terrestrial networks brings significant advantages in terms of network effectiveness. We show that elastic buffering is an extremely effective means to reduce, or even eliminate, streaming service outage due to intermittent backbone connectivity, such as that occurring when a vehicle moves through tunnels. Moreover, we show that elastic buffering is not only a technique suitable for multimedia information retrieval services, but can be effectively applied to delayed real-time services.

INTRODUCTION

Mobile networking not only tackles the problem of providing a networking infrastructure for mobile customers, but also includes the ability to manage moving networks. This is the case with vehicular area networks (VANs). These networks are formed by customers located on the same moving vehicle (e.g., a moving train or bus). The moving network is interconnected to the terrestrial network via one or more wireless links. Satellite links are frequently considered an important candidate technology in practical deployment scenarios, encompassing public trains, ships, airplanes, buses, and so on. But emerging wireless technologies for metropolitan high-speed networks, such as IEEE 802.16 and IEEE 802.20, may become in the near future important technologies in the VAN arena.

The scenario considered here is that of a VAN in which users' connectivity to the rest of the network is managed by a specialized onboard gateway. The role of this gateway is to provide internetworking between the internal network and wireless technology adopted for connecting to the terrestrial network. For sake of simplicity, in this article we assume that such a connection is accomplished through a single high-capacity wireless link (e.g. a satellite link), hereafter referred to as wireless backbone. The network architecture deployed within the VAN can be built either on wireless or wired LAN technologies, its implementation details being out of the scope of this article, and depending on the considered vehicular network application scenario.

In addition to internetworking functions, the gateway may act as proxy server. As such, it may provide local storage capabilities to support caching and/or prefetching algorithms. These mechanisms are meant to support video and interactive streams [1, 2], as well as multimedia with quality of service [3, 4]. These mechanisms are also devised to maximize the probability that a customer requesting an object to download (e.g., a Web page or video segment) may find it stored in a repository associated with the proxy, thus improving retrieval performance and reducing the traffic load on the wireless backbone. Effective caching and prefetching mechanisms for streaming services, which may be applied with minimal or null modification to the VAN scenario considered in this work, have been thoroughly studied in [5–7], also considering prefetching located on the edge of the backbone network, in edge routers, where a microcell or overlying network is present, as in [8].

A problem typical of the VAN scenario is the possible outage of the wireless backbone. Such an outage, hereafter referred to as *channel outage*, may occur while the vehicle crosses through areas characterized by severe fading conditions (e.g., tunnels). This is a very critical issue when dealing with streaming services, which experience possibly long (on the order of several seconds) interruptions, thus causing highly negative performance impairments (service disruption) in terms of the end customer point of view. In what follows we refer to an event of service interruption as connection outage.

The proxy solution is also suitable in order to avoid the end user dealing with handover issues, since the mobile proxy could be made able to

This research is partially supported by the Italian Ministry of Research in the frame of the FIRB VICOM project, and by the European Community in the frame of the IST FIFTH Project

0163-6804/04/$20.00 © 2004 IEEE

take care of eventually needed handover (this is still not true in the case of satellite, but in general it could be true if one uses wireless MAN connectivity standards); for example, a proxy could implement techniques devised in [9], where prefetching issues are also addressed.

Now, we argue that, other than traditional caching and prefetching mechanisms, a further role of the proxy is to support mechanisms devised to hide channel outage periods from the final user (i.e., reduce the impact of channel outage in terms of resulting connection outage). This can be accomplished by decoupling, from a service level point of view, the delivery service provided inside the moving network from that enforced on the wireless backbone. Specifically, the presence of an onboard proxy enables us to split the streaming service into two independent parts. Within the moving network, the stream is delivered to the end user at its natural playout speed. For convenience of presentation, in what follows we assume a constant playout rate.[1] Conversely, multimedia information may be retrieved on the wireless backbone at a variable speed, eventually higher than the playout rate; the resulting excess information retrieved is accumulated in a proxy buffer for future playout. Hence, the per-stream proxy buffer can be seen as an "elastic" buffer that empties at a constant rate and fills at variable rate: although during channel outage no information can be retrieved, this does not automatically lead to connection outage, as long as sufficient information has been previously stored in the buffer.

The rest of this article is organized as follows. We review the role of elastic buffering in wired networks [10] and its highly beneficial effects in terms of network efficiency. We tackle the moving network and channel outage scenario, and propose a resource management mechanism specifically designed for such a scenario. While the focus of that section is on multimedia information retrieval (e.g., video on demand) services, in the next section we show that the same concepts can be adapted to provide more effective support for broadcast delayed services, that is, delayed access to broadcast multimedia transmission (e.g., television channels). Performance investigation is carried out for both multimedia on demand services and delayed real-time services. Finally, concluding remarks are given.

Performance Effectiveness of Elastic Proxy Buffer Management

Before tackling the issue of moving networks, let us first review the fundamental performance advantages provided by the adoption of elastic buffering mechanisms in support of streaming services. To the authors' knowledge, our early work [10] is the first article to tackle this issue.

The basic idea is very simple. Without a proxy, a multimedia streaming session is set up directly between the end user and a remote streaming server. Assuming limited buffering capabilities at the receiver, the delivery rate of the multimedia information is of course equal to its playout rate at the receiver. Let us now insert into this delivery process an intermediate dynamic storage capability (i.e., a buffer) placed on a proxy between the receiver and the video server. It is thus possible to split the streaming connection into two parts. From the proxy to the end user, the multimedia information will be delivered at its natural playout speed. But from the server to the proxy, it is possible to retrieve the multimedia information at a rate higher than the playout rate. The excess information downloaded from the satellite is in fact temporarily buffered in the storage resources available at the proxy.

This operation leads to significant improvement in network utilization. As shown in [10, Sec. 5], such a performance advantage can be analytically quantified under the assumption of infinite buffer size, and considering a simplified network scenario composed of a single network link, referred to in what follows the backbone, connecting the proxy to the server.

In fact, let b b/s be the playout speed of multimedia streaming sessions (assumed homogeneous). Let B b/s be the backbone capacity. For convenience, let B be a multiple of b. Hence, $K = B/b$ represents the maximum number of simultaneous streaming sessions that can be supported on the backbone. Assume a Poisson arrival process of streaming session requests, with rate λ requests/s, and let $E[T]$ be the mean playout duration of the multimedia streams (what follows holds for general distribution of the stream duration, i.e., the result depends only on the mean value $E[T]$). Let $r = \lambda E[T]$. Then in stable conditions (i.e., $r < K$):

- Without a proxy buffer, the number of simultaneous streams carried on the backbone obviously follows a Poisson distribution; that is, the probability of finding n streams on the backbone is $e^{-r}r^n/(n!)$.
- With the intermediate proxy buffer, the backbone can instead be modeled as an $M/G/1$-PS processor sharing queuing system, where the probability of finding n streams on the backbone is $(1 - r/K)(r/K)^n$.

In [10] a comparison between these two cases is given in terms of backbone blocking performance,[2] assuming that an admission control rule is deployed on the backbone so that no more than K concurrent sessions can be simultaneously admitted. Without a proxy buffer, the request blocking probability is readily computed via the Erlang-B formula $B(r,K)$. With a proxy buffer, the blocking probability is computed as the drop ratio of an $M/M/1/K$ finite size queuing system (i.e., $(1 - r/K)(r/K)^K/[1 - (r/K)^{K+1}]$). The elastic buffer solution outperforms the standard streaming approach (for a numerical example, the reader may easily compute that when $r = 85$ and $K = 100$, the blocking probability experienced by users reduces from about 10^{-2} to about 10^{-8} with an elastic buffer). Simulation results obtained with finite buffer size [11] show that significant performance advantages can be obtained even with fairly limited storage availability. Finally, more efficient buffer management policies, integrating elastic buffering with traditional caching, are presented in [10].

[1] In fact, the basic ideas proposed in this article do not require a constant playout rate. However, the detailed design of the resource management algorithm on the wireless backbone may slightly differ in the case of variable-rate streaming services from that proposed in this article, as it might be optimized to further take advantage of knowledge of the traffic flow statistical description.

[2] We note that the discussion carried out in [10] was focused on an ATM transport network scenario, that technology being fully operational at the moment of writing that article. In the ATM framework, it was natural to deploy the proposed approach on the available bit rate (ABR) service, set the minimum cell rate (MCR) to the playout rate of the stream, and evaluate the performance in terms of connection blocking. The performance insights given in this section suggest also pushing toward deployment of elastic buffering mechanisms for streaming service support over the Internet, using TCP on the network core rather than constant-rate streaming over RTP/UDP. Of course, unlike the case of ATM, an admission control rule suitable for this scenario and therefore acting on TCP flows is harder to deploy.

By enforcing an elastic buffer management in the on-board proxy, multimedia streaming sessions may remain active even during satellite outage periods, provided that a sufficient amount of information has been buffered in the proxy local storage.

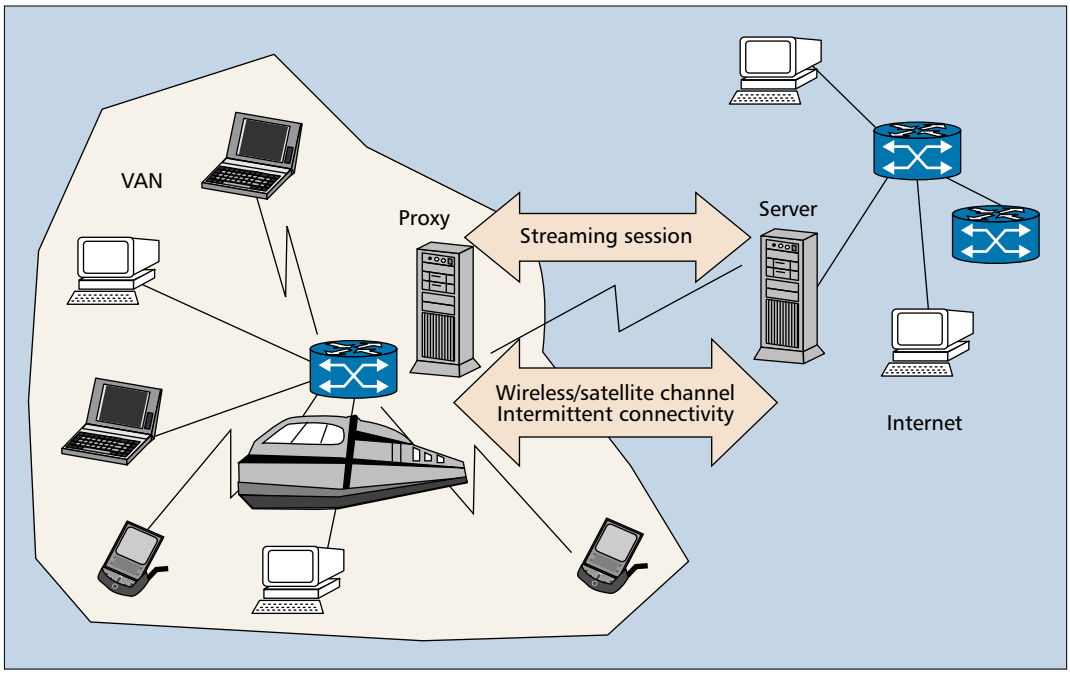

Figure 1. *The vehicular area network scenario.*

Adaptation of Elastic Buffering to Moving Networks

In addition to the role, discussed in the previous section, of improving the network performance, in the rest of this article we show that in a moving network scenario, elastic buffering brings additional significant advantages.

Figure 1 illustrates the concept of a VAN. In traditional wireless networks, each user sitting on a moving vehicle connects independently to the terrestrial network. Conversely, in a VAN scenario, a network — typically a LAN or wireless LAN (WLAN) — is deployed inside a moving vehicle. End-user connectivity to the terrestrial network (i.e., the Internet) is managed through a specialized onboard gateway. We propose to endow the onboard gateway with proxy functionalities.

In what follows we illustrate how a VAN can take advantage of an elastic buffer mechanism implemented on the onboard proxy. As long as some minimum requirements are met, the described ideas do not specifically depend on the networking technology employed inside the VAN, the wireless backbone technology, and the applications scenario. However, to provide a more concrete presentation, we expose our basic ideas with reference to a particular application scenario: a train network connected to the terrestrial network through a satellite link.[3]

The goal of our approach is to make the system robust to the uncertain behavior of the satellite link. In fact, the satellite link is characterized by a wideband high-performing channel, but it requires a line-of-sight connection. During satellite link outage, which may last for several seconds (e.g., while crossing tunnels), all communications are interrupted since no information can be received at the onboard gateway and, in turn, routed toward the relevant end user.

Let us first focus on multimedia on demand services (extension to delayed real-time services will be tackled in the next section). By enforcing an elastic buffer management in the onboard proxy, multimedia streaming sessions may remain active even during satellite outage periods, provided a sufficient amount of information has been buffered in the proxy local storage. The streaming session presents outage (connection outage) only when both the satellite link is out and the buffered information is exhausted.

We show in the following subsection that a very important role is played by the strategy adopted in managing satellite link capacity. Specifically, uniform allocation of the extra available bandwidth to concurrent downloads, while effective in wired networks (as reviewed earlier), is a poor strategy in the VAN scenario.

The All-To-Min Resource Sharing Algorithm

The described operation gives rise to an elastic buffer, which is filled during the periods in which the satellite link is active, and whose buffered data is consumed at a fixed rate given by the playout speed for each streaming session, multiplied by the number of concurrent streaming sessions.

For the sake of simplicity, assume that the multimedia information repository is directly placed at the terrestrial satellite gateway. Let us consider an onboard customer requesting a file. Once the streaming request is received at the proxy, it first checks if the requested stream is locally cached. If not, the request is forwarded to the remote server, provided satellite resources are available (i.e., after a positive response of an admission control decision based on the number of already active streaming sessions on the satellite link). All the satellite link capacity is shared

[3] *Indeed, this is the application scenario tackled in a European Community funded IST project, called FIFTH: Fast Internet for Fast Train Hosts. Other VAN application scenarios are currently being considered, in either research projects (e.g., ships are tackled in the IST project MOBILITY) or commercial deployment (airplanes and buses). But the case of trains and buses is the one that presents major challenges at the data link and network layers, due to the occurrence of tunnel crossing, which is not applicable to ships and airplanes.*

between all active sessions, so the multimedia information is retrieved on the satellite link at the maximum possible rate.

Figure 2 describes two different resource sharing policies that can be enforced on the satellite link. The simplest approach is to fairly share the satellite resources among all concurrent multimedia information retrievals. This policy is referred to as *equally distributed* (ED). Figure 2 illustrates ED operation (solid lines) by reporting the per-session proxy buffer occupancy level vs. time. It is assumed that at time 0 a new connection, session #1, starts. Let C b/s be the satellite channel capacity, and R b/s the streaming playout rate (assumed equal for all sessions). Since a single session is offered to the satellite link, all the channel capacity C is reserved to download information. This information is in turn played out at rate R. We conclude that the buffer level grows linearly with time, $(C - R)t_1$ bits being the buffer-level at time t_1. Assume now that at time t_1 session #2 starts. At time t_2 the buffered data for this latter session will amount to $(C/2 - R)(t_2 - t_1)$, while the level of the buffer occupation for session #1 will have reached $(C - R)t_1 + (C/2 - R)(t_2 - t_1)$.

In general, when n sessions share the link, each is granted a retrieval rate equal to C/n. If no channel outage is assumed to occur, a reasonable admission control rule is to admit a new session as long as C/n remains greater than or equal to R (ED is in fact the policy assumed in the discussion earlier, and this was the related admission control rule).

A closer look at Fig. 2 allows us to underline a major limit of the ED policy. In fact, whenever a channel outage occurs (in the figure, from time t_{outage} to time t_{end}), the most recently admitted sessions are the first ones for which connection outage occurs. In turn, the buffer level reached by the first admitted flow is unnecessarily high. In other words, fair resource sharing on the channel yields an unfair share of the proxy buffer space and increases the probability of connection outage occurring during a channel outage period.

In the case of channel outage, the buffer level represents the margin before an outage occurs. For this reason we also use the term *outage margin*. It naturally comes out that an effective resource sharing approach is to devise a policy targeted to converge as fast as possible to a situation in which all sessions have the same outage margin.

We refer to such a new policy by the name *all to minimum* (A2M). The A2M operation is designed to dynamically reserve all the channel resources to the session(s) with lower buffer level(s). This operation is implemented at the proxy, which can access the information regarding the per-flow buffered data. In turn, the proxy dynamically signals (through layer 2 signaling mechanisms, e.g., satellite-specific mechanisms) to the terrestrial gateway the identity of the sessions that suffer at minimum buffer level, so this latter gateway is able to properly schedule the information download.

The A2M operation is exemplified in Fig. 2 (dashed lines). At time 0, the A2M algorithm operates just as ED does, since a single session

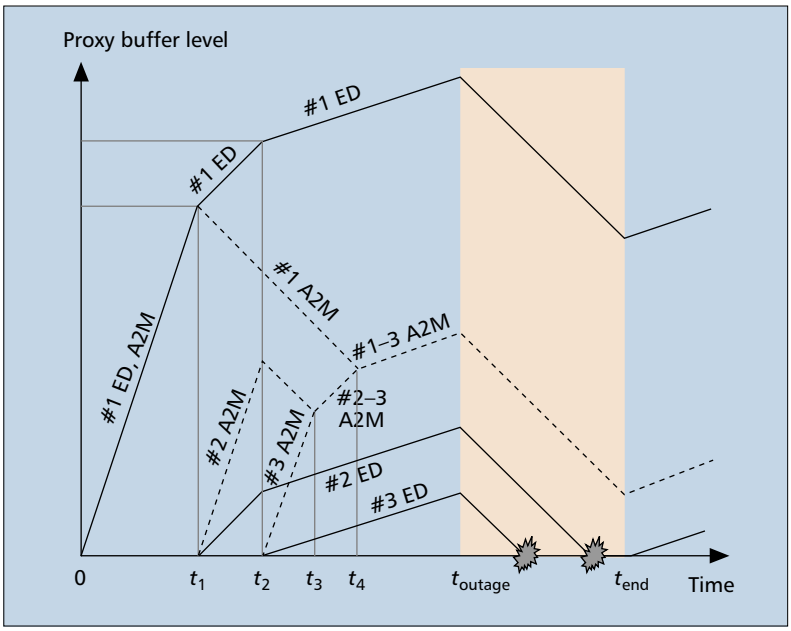

Figure 2. *A2M vs. ED operation.*

is admitted on the channel. The differences start from the instant of time t_1 in which session #2 starts. In fact, from time t_1 all the channel capacity is reserved to session #2. Hence, its buffer level grows at rate $(C - R)$. Conversely, since no channel resources are assigned to session #1, its buffer level decreases at rate R. This lasts until the two buffer levels become the same. If, in the meantime, a new session #3 starts (time t_2 in the figure), it receives all the channel resources until it reaches the same buffer level as session #2 (this occurs at time t_3 in the figure). Then channel capacity C is equally shared between sessions #2 and #3, so their buffer level grows at rate $(C/2 - R)$ while the buffer level of session #1 decreases at rate R. Once all n (three in this example) per-session buffered data have reached the same level (time t_4), a uniform share C/n of channel capacity will be enforced again.

As is apparent from the figure, the A2M mechanism minimizes the probability of a connection outage occurring during a channel outage period. The price to pay is that if a connection outage occurs, it is likely to simultaneously involve all the admitted sessions.

The A2M algorithm can easily be extended to a framework where streaming sessions are characterized by different data rates. The only difference is that the per-session buffer level has to be measured in terms of playout time (i.e., time to reproduce content on a user's terminal).

Adaptation to Delayed Real-Time Streaming Services

Multimedia on demand services are hardly supported in a scenario characterized by a very large number of customers, such as train passengers, and fairly limited wireless backbone bandwidth, such as on the satellite link. In fact, the satellite link may support only a limited amount

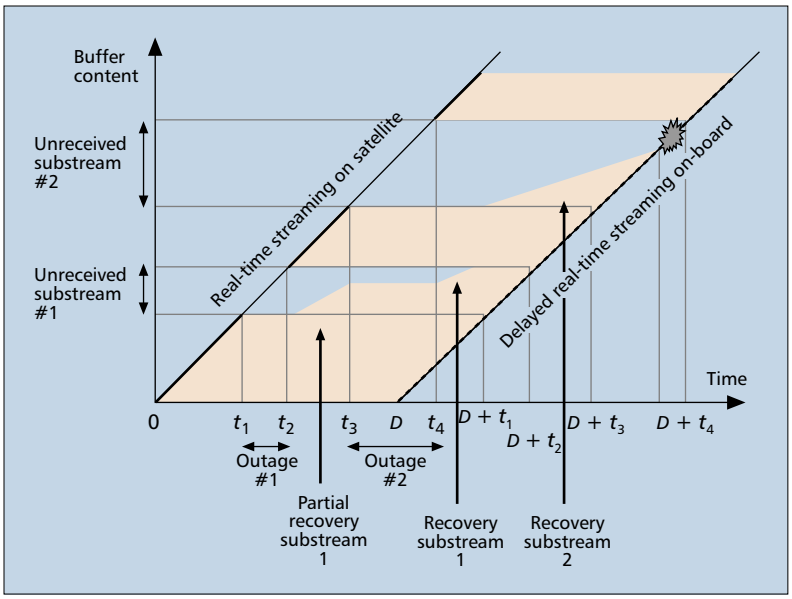

Figure 3. *Connection outage after multiple channel outages.*

of simultaneous streams, and thus only very few customers may receive service. Conversely, seamless support of broadcast streaming services (e.g., digital television channels) on moving networks is an extremely appealing challenge for a public vehicle operator.

Of course, due to the presence of tunnels it is possible to compensate channel outage for real-time streams only if their delivery inside the train is delayed for a time greater than the crossing time of the longest tunnel along the train path. In other words, if an event is scheduled to be broadcast at a given time t, onboard delivery will start at time $t + D$, and the proxy will buffer all the information related to the D s of delay introduced. Delayed playout can be accomplished by simply using fixed buffers whose size is exactly equal to that needed to store D s of broadcast video. Of course, when an outage lasting $\delta > D$ s occurs on the satellite link, the proxy will be able to play out only the first D s of the considered stream, while connection outage will occur for the remaining $\delta - D$ s.

However, a closer look at this problem reveals that an initial playout delay greater than the crossing time of the longest tunnel along the train path is not sufficient by itself to avoid outage. This is evident considering the example depicted in Fig. 3. This figure reports real-time streaming received from the satellite (solid line) and delayed real-time streaming delivery occurring onboard (dashed line). The x-axis reports the elapsed time. As shown in the figure, the delayed playout starts with an initial offset equal to D. Two channel outage periods are illustrated in the figure: one occurring in the range (t_1, t_2) and the other in the range (t_3, t_4). If no action is taken, it is evident that during an outage period, a fraction of the original stream (referred to as substreams #1 and #2 in the figure) will not be stored in the proxy buffer. Hence, after delay D, this outage will also be experienced within the train, and specifically in the time ranges $(D + t_1, D + t_2)$ and $(D + t_3, D + t_4)$. In other words, the difference with respect to the multimedia information retrieval scenario stays in the fact that during a channel outage, the satellite broadcasting goes on; hence, the missing (unreceived) substreams cannot be temporarily stored in the buffer for subsequent playout.

We propose to solve the above problem by treating each substream as an independent special case of a multimedia information retrieval session. As soon as an outage period ends, the proxy buffer uses extra bandwidth available on the satellite channel to recover the substream, and thus refill the "buffer hole" caused by the channel outage. This implies that to effectively support delayed real-time services, the channel capacity must be split in two parts: one to transmit the real-time streams, and the other for the described substream recovery procedure.

It will be shown in the numerical results section that the extra bandwidth made available to delayed real-time services must be quite large. Since, in addition, this extra bandwidth is used only at the end of a channel outage, and generally for a short time (i.e., in very bursty mode), it is convenient to let the substream recovery procedures share the same bandwidth reserved for multimedia on demand streaming sessions. In doing this, it is worth noting that the A2M algorithm can be adopted to manage this extra bandwidth; that is, in principle no distinction is needed between substream recovery procedures and normal on-demand multimedia streaming support.

To better understand how connection outage may arise, consider again the example illustrated in Fig. 3. This is a scenario in which recovery of multiple unreceived substreams may occur. In fact, the recovery phase for substream #1 starts at time t_2, right after the end of the relevant outage period. However, due to scarce extra bandwidth availability in the time interval (t_2, t_3) and/or too short a time range (t_2, t_3) elapsing before outage #2 occurs, it is possible that only part of substream #1 is recovered. As illustrated in the figure, at time t_4 the recovery procedure for substream #1 restarts (in fact, following the A2M rules defined earlier, the outage margin for substream #1 is lower than that for substream #2). Hence, all the available bandwidth is assigned to substream #1 until its complete recovery. But this delays the start of the recovery procedure for substream #2. In other words, even if the outage margin after channel outage #2 was large enough to allow full recovery of substream #2, the presence of an additional substream to be recovered leads to connection outage.

An important parameter is the ratio K between the spare bandwidth made available for recovery and the delayed streaming service rate. This parameter can be intended as the amount of video seconds that will be recovered in 1 s. In practice, when $K > 1$, connection outage can occur only when the train is in a tunnel. But $K > 1$ means that more than 50 percent of total bandwidth is allotted to recovery operations; this is poor link utilization efficiency. When $K < 1$, the time needed to recover a substream is longer

than the substream itself; this means that it cannot be recovered while being delivering, and an outage occurs.

Performance Evaluation

The effectiveness introduced by the A2M approach compared to the ED mechanism is evaluated in this section. Specifically, the next section deals with multimedia on demand services, while the case of delayed real-time (RT) services is discussed next. Finally, simulation results taken from a train scenario currently under deployment are presented.

Multimedia on Demand Services

We have considered a scenario characterized by homogeneous multimedia on demand streaming sessions, each requiring a constant playout rate $R = 1$ Mb/s. The wireless backbone capacity has been set to $C = 16$ Mb/s. We assume that each streaming session lasts for an amount of time fixed to 1800 s (the time needed for the internal network to complete stream delivery, i.e., 1800 s is the stream duration at playout time).

To stress the wireless backbone, we have assumed a load regime referred to in what follows as *saturation load*. Specifically, the number of simultaneous retrievals on the wireless backbone has been enforced not to go over a maximum threshold (eight retrievals in the simulation runs). As soon as a retrieval terminates, we assume that a new streaming request is immediately available. We remark that this is a special case of dynamic network operation, where new requests arrive at a very high (infinite) arrival rate. Numerical results obtained for Poisson arrival of requests and finite rate, and not shown here for reasons of space, show that the saturation load represents the worst-case condition in terms of wireless backbone performance. We also remark that the number of users served, on average, in saturation load conditions is actually greater than the above mentioned threshold, as a new streaming session starts as long as the previous streaming session has completed information retrieval (but this can happen well before the stream delivery in the internal network ends).

Channel outage has been modeled through an ON-OFF pattern, meaning that during an outage period (i.e., inside a tunnel) no connectivity is possible, while connectivity at full capacity C is provided in visibility. We have assumed outage periods (tunnels) to have an exponentially distributed duration, with mean value 180 s. The duration of the ON periods is set depending on the channel outage probability target $P_{ch} = T_{OFF}/(T_{ON} + T_{OFF})$.

The proposed scenario has been tested by means of a fluidic C++ event-driven simulator. In the simulation resource management on the wireless backbone is assumed to be instantaneous. The proxy buffer space is assumed to be infinite (although in practice its occupation level always remains bounded). In a satellite network scenario, resource allocation will be managed at the data link layer, and additional management delays arise. However, we remark that even for geostationary satellites, such signaling delays are negligible compared to the time needed to go through a tunnel.

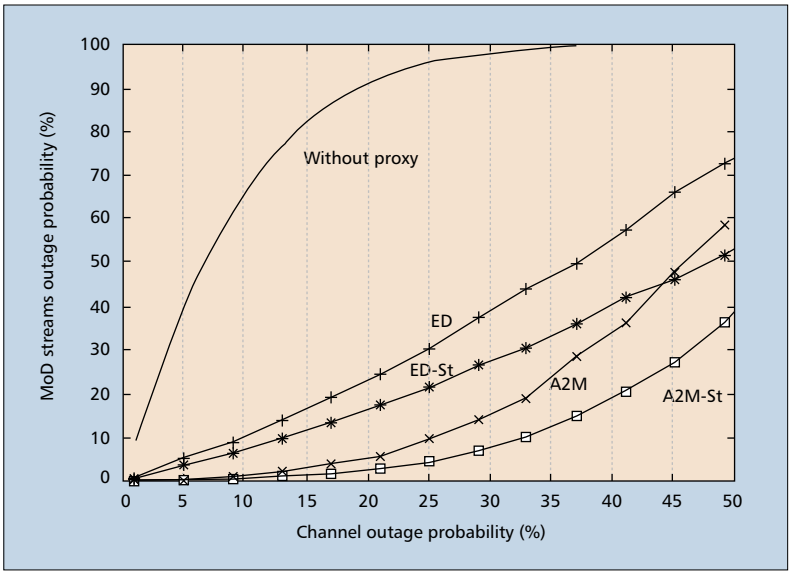

Figure 4. *Multimedia on demand: connection outage probability vs. channel outage probability.*

Figure 4 shows the ED and A2M performance in terms of connection outage probability vs. channel outage probability. In particular, connection outage probability is defined as the probability that connection outage occurs at least one time during the stream duration. As shown in the figure, if no proxy were employed, this probability would rapidly converge to 100 percent as the channel outage probability grows. We see that the improvement provided by elastic buffering is remarkable. Moreover, A2M significantly outperforms ED in all cases. Regarding buffer space consumption, it is very interesting, and perhaps not intuitive, to note that the buffer space needed on the proxy is very limited. Numerical results obtained during the 50 percent channel outage probability simulation run show an average buffer occupancy level of about 2500 s (i.e., less than one and a half stream size), while this value decreases to as low as 500 s in the 75 percent channel outage probability case (clearly, the more severe the outage, the less loaded the buffers).

To demonstrate that the proposed proxy management scheme can be efficiently run in conjunction with a caching or prefetching mechanism, the figure reports the performance of the ED and A2M schemes (labeled ED-St and A2M-St) under the assumption that 30 percent of the incoming requests match files preloaded in the proxy memory for half of their size (see [1, 7] for insights on partial prefetching schemes). As expected, prefetching leads to performance improvement, but what is interesting from our point of view is that A2M shows a further relative advantage over ED.

Figure 4 is limited to present the probability that at least one outage occurs. To complement these results, Fig. 5 reports the probability distribution of the number of outages experienced by a single stream. A 25 percent channel outage probability and the A2M mechanism are consid-

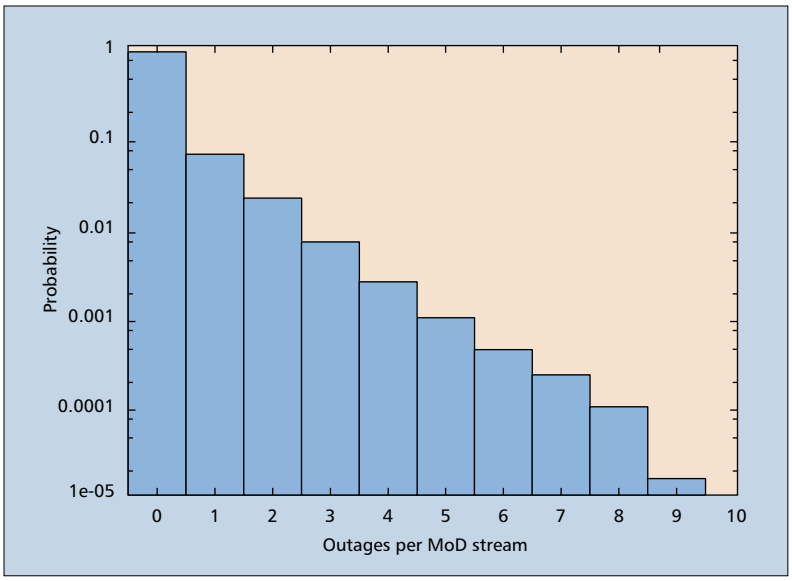

Figure 5. *Probability that a multimedia on demand stream experiences #n outages, with A2M.*

ered in this figure. As shown by the figure (note the y-axis log scale), the probability that a stream experiences two or more outages is very low.

Delayed Real-Time Services

Performance results concerning the support of delayed real-time services are reported in Fig. 6. Results were obtained by means of the same C++ simulation cited in the previous section, using delayed real-time streams instead of multimedia on demand ones.

The wireless backbone was loaded with a single everlasting broadcast transmission. In order to present cleaner results, we have not included multimedia on demand sessions in this scenario.

The two major design parameters to be considered in such simulations are the playout delay to be adopted before delivering a stream to the end user, and the amount of extra bandwidth resources reserved for recovery procedures.

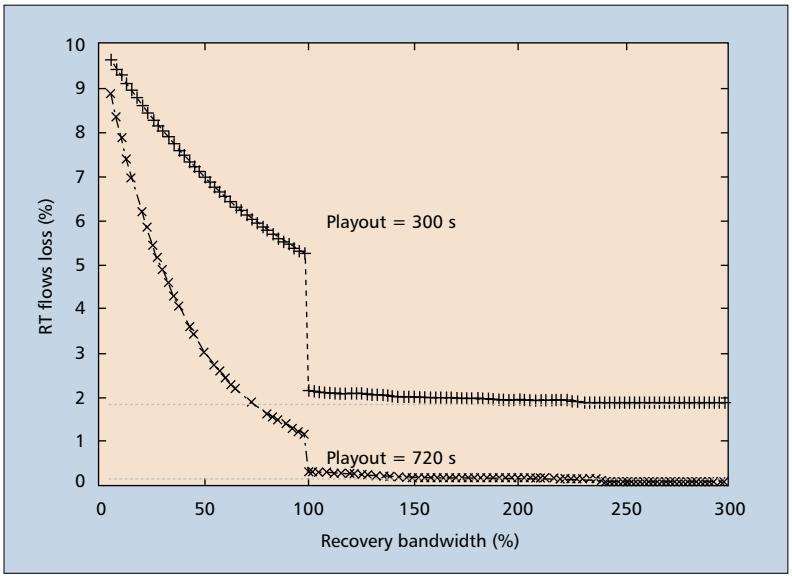

Figure 6. *Delayed real time stream: fraction of unrecovered (lost) data.*

Figure 6 reports the percentage of real-time stream lost as a consequence of connection outage (unrecovered data) vs. the recovery bandwidth (i.e., the extra bandwidth with respect to the natural real-time broadcast rate). Two initial playout delays are considered: 300 s and 720 s. The channel outage has been modeled by assuming exponentially distributed outage periods lasting, on average, 180 s and exponentially distributed visibility periods with mean value 1620 s (i.e., a 10 percent channel outage probability). The figure shows that as long as the recovery bandwidth increases, the performance improves. The amount of the lost fraction of the stream converges to an asymptotic value which simply represents the probability that a single tunnel (i.e., an outage period) lasts more than the initial playout delay; in such a case, a fraction of the stream will be lost regardless of eventually unlimited extra bandwidth available.

A sharp improvement in performance is suddenly encountered when the extra bandwidth available is equal to the stream rate. This operational point corresponds to $K = 1$, as defined earlier. In fact, for $K \geq 1$, the recovery bandwidth is greater than or equal to the stream rate; thus, connection outage can occur only inside a tunnel. Conversely, when $K < 1$, the recovery rate (the rate at which the proxy buffer is filled) is lower than the stream rate (the rate at which the proxy buffer is emptied); thus, connection outage may occur outside a tunnel as long as the buffered information exhausts (see also Fig. 3).

The Train Scenario: Italian Railways

In this section we present performance results taken from the reference deployment scenario tackled in the European Community IST project Fast Internet for Fast Train Hosts (FIFTH). The goal of this project is to deploy satellite communication for providing Internet and multimedia streaming services aboard high-speed trains. The following figures report results for a deterministic tunnel pattern taken on the railway path between the Italian cities Rome and Florence (covered in about 1.5 h by high-speed trains). About 23 percent of this path is covered by 44 tunnels, the longest lasting for about 180 s. The average passing time through a tunnel is 24 s, although the tunnel sizes show a "bursty" pattern, and tunnels appear to cluster.

Figure 7 presents results for multimedia on demand services. It assumes eight streams in saturation load, and shows the relationship between the extra bandwidth to be made available on the satellite channel vs. the extra time needed to complete the vision of the streams. We have reported such a performance figure as it provides a thorough insight in terms of exploitation: during a connection outage, advertising might be delivered to customers, and the figure may be intended to quantify how much advertising overhead a customer shall suffer vs. a given satellite bandwidth dimensioning. From the figure it appears that a small extra bandwidth assignment is shown to be able to drastically reduce the download time. For example, a 20 percent extra sizing of the satellite bandwidth leads to about

an extra 13 percent of the stream delivery time. However, the figure shows that to achieve a marginal completion time overhead, considerable satellite bandwidth overprovisioning (up to 70 percent and more) is required.

Finally, Fig. 8 reports results for a delayed broadcast (real-time) stream. Here, the obvious service requirement for deployment purposes is to provide seamless support of the broadcast channel (i.e., design the system so that no outage periods would occur). The plot reported in the figure shows the initial playout delay (y-axis) and the extra bandwidth sizing (x-axis) necessary to achieve uninterrupted service (zero-loss). The figure clearly shows that initial playout delay can be traded off with extra bandwidth. It also shows that a minimum initial playout delay equal to the longest involved tunnel (about 180 s) is necessary to provide uninterrupted service support.

Conclusions and Open Issues

In this article we have reviewed the advantages provided by the adoption of elastic buffering in support of multimedia streaming. Elastic buffering relies on a proxy devised to split data flows into two parts. The proxy-to-client stream is delivered at the natural playout rate. Conversely, the stream is downloaded from the remote server to the proxy at a rate possibly higher than playout. Excess retrieved information is temporarily stored in the proxy buffer for future delivery. With focus on vehicular area networks, we have tackled the problem of designing a suitable mechanism to manage resources on the wireless backbone. Two different service scenarios have been considered: multimedia on demand streaming sessions and delayed real-time services. Simulation results show the effectiveness of elastic buffering to compensate for intermittent connectivity occurring during outage periods. Moreover, they show the superiority of the proposed A2M mechanism with respect to a "traditional" approach in which the bandwidth on the wireless backbone is equally shared among active streams.

This article leaves open the issue of how to deploy the proposed mechanisms over the Internet. Such a deployment requires thorough investigation of signaling mechanisms between the proxy and the remote server for dynamic support of play/pause/resume functions implemented at the proxy side, and their application on top of a TCP session between proxy and remote server to achieve information download at a speed higher than the natural streaming playout.

Finally, the basic ideas proposed in this article are being developed within the framework of the European Community funded IST FIFTH project, which employs a bidirectional communication between a geostationary Earth orbit satellite and a traveling train equipped with an auto-seeking onboard parabolic antenna. The investigation carried out for this scenario has assumed a dedicated satellite connection for the moving train, as this will be the demonstration target for the project. Further performance insights are needed to understand the effectiveness of the proposed approach in a scenario where a fleet of trains sharing the same satellite link needs to be networked.

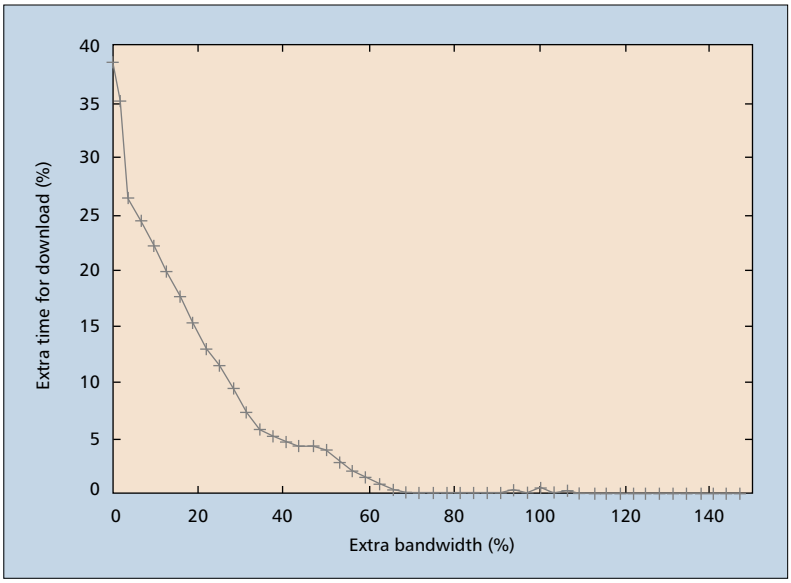

Figure 7. *The Rome–Florence railway path: multimedia on demand performance.*

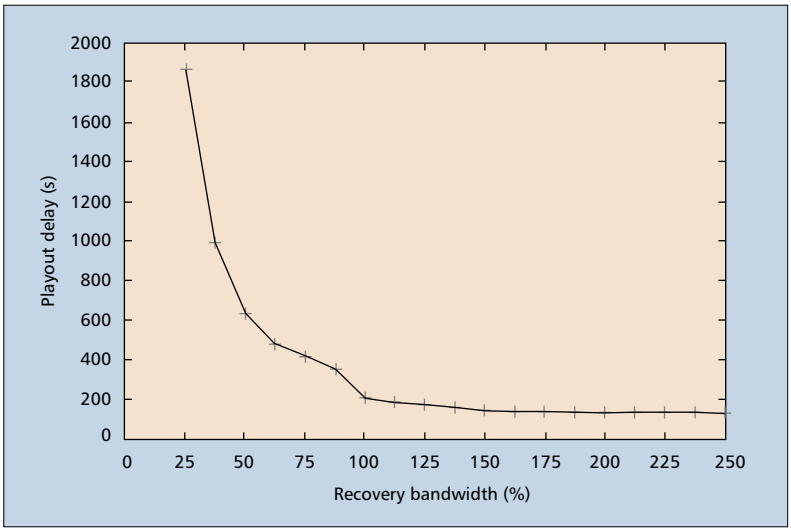

Figure 8. *The Rome–Florence railway path: delayed real-time zero loss tradeoffs.*

References

[1] S. Sent, J. Rexford, and D. Towsley, "Proxy Prefix Caching for Multimedia Streams," *Proc. INFOCOM '99*, Apr. 1999.

[2] M. Reissline, F. Hartanto, and K. W. Ross, "Interactive Video Streaming with Proxy Servers," *Proc. IMMCN*, Feb. 2000.

[3] R. Rejaie *et al.*, "Multimedia Proxy Caching Mechanisms for Quality Adaptive Streaming Applications in the Internet," *Proc. INFOCOM 2000*, Mar. 2000.

[4] B. Wang *et al.*, "Optimal Proxy Cache Allocation for Efficient Streaming Media Distribution," *Proc. INFOCOM '02*, 2002

[5] Y. Wang *et al.*, "A Network-Conscious Approach of End-to-End Video Delivery over Wide Area Networks Using Proxy Servers," *Proc. INFOCOM '98*, 1998

[6] M. Crovella and P. Barford, "The Network Effects of Prefetching," *Proc. IEEE INFOCOM '98*, 1998

[7] S. Jin, A. Bestavros, and A. Iyengar, "Accelerating Internet Streaming Media Delivery Using Network-Aware Partial Caching," *Proc. 22th Int'l. Conf. Comp. Sys.*, 2002.

[8] N. Imai et al., "On On-demand Data Prefetching System for Spotted Access Networks," *IEICE Trans. Commun.*, vol. E84-B, no. 10, Oct. 2001
[9] M. Stemm and R. H. Katz, "Vertical Handoffs in Wireless Overlay Networks," *ACM/Baltzer Mobile Net. and Apps.*, Special Issue on Mobile Networking in the Internet, vol. 3, no. 4, Jan. 1999, pp. 319–34.
[10] G. Bianchi and R. Melen, "The Role of the Local Storage in Supporting Video Retrieval Services on ATM Networks," *IEEE/ACM Trans. Net.*, vol. 5, no.6, Dec. 1997.
[11] G. Bianchi, "Buffer Sizing for High Speed Video Information Retrieval on ATM Networks," *Proc. IEEE GLOBECOM '97*, Phoenix, AZ, Nov. 1997.

BIOGRAPHIES

VINCENZO MANCUSO (vincenzo.mancuso@tti.unipa.it) received a Laurea degree in electronic engineering, magna cum laude, from the University of Palermo, Italy, in 2001. He is now a candidate for the Ph.D. in telecommunications at the University of Palermo. His research interests include QoS mechanisms for stateless networks, and multimedia and real time services support in heterogeneous networks.

GIUSEPPE BIANCHI (bianchi@elet.polimi.it) received a Laurea degree in electronic engineering from the Polytechnic of Milano, Italy, in 1990, and a specialization degree in information technology from Cefriel, Milano, in 1991. He spent 1992 as a visiting researcher at Washington University, St. Louis, Missouri, and 1997 as a visiting professor at Columbia University, New York. He was an assistant professor at the Polytechnic of Milano from 1993 to 1998 and an associate professor at the University of Palermo from 1998 to 2003. He is currently an associate professor at the University of Roma Tor Vergata. His research interests include wireless access protocols and network architectures, QoS support in both wireless and wired IP networks, and performance evaluation.