

# Speeding up the Evaluation of Multimedia Streaming Applications in MANETs using HMMs \*

Carlos T. Calafate  
calafate@disca.upv.es

Pietro Manzoni  
pmanzoni@disca.upv.es

Manuel P. Malumbres  
mperez@disca.upv.es

Department of Computer Engineering  
Polytechnic University of Valencia (UPV)  
Valencia, Spain

## ABSTRACT

Mobile ad-hoc networks (MANETs) present quite large packet loss bursts due to mobility. In this work we propose two models based on hidden Markov models for estimating packet arrivals and packet loss patterns in MANETs. These models help the evaluation and tuning of multimedia streaming applications in terms of simulation time and required resources. In particular, we show how these models can be applied in the design of error concealment algorithms to increase the video coding resilience. The obtained results show that we get comparable results without the need for several long simulation runs. Finally, we also propose a set of new metrics for packet loss patterns analysis that can be of interest for the evaluation of audio/video streaming applications.

## Categories and Subject Descriptors

I.6.4 [Simulation and Modeling]: Model Validation and Analysis

## General Terms

Verification, Performance

## Keywords

Wireless ad-hoc networks, Markov models, burst metrics

## 1. INTRODUCTION

Mobile ad-hoc networks (MANET) [1] are wireless networks with no fixed infrastructure. Nodes belonging to a MANET can either be end-points of a data interchange or can act as routers when the two end-points are not directly within their radio range. The most widely deployed technology to implement this kind of networks is based on the IEEE 802.11 [2] standard. Wireless ad-hoc networks suffer from frequent topology changes and provide a poor

\*This work was supported by the *Ministerio de Ciencia y Tecnología under grant TIC/2003/00339*, and the *Junta de Comunidades de Castilla La-Mancha under grant PBC/03/001*.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

MSWiM'04, October 4-6, 2004, Venezia, Italy.

Copyright 2004 ACM 1-58113-953-5/04/0010 ...\$5.00.

QoS support. However, support for real-time communication in wireless networks is becoming more and more important due to the increasing demand for multimedia applications.

The issue of topology variability can only be handled through efficient routing mechanisms. A couple of years ago near to 60 proposals of routing protocols were being evaluated. Nowadays only four proposals, respectively the "Ad hoc On Demand Distance Vector" (AODV) [3], the "Dynamic Source Routing Protocol for Mobile Ad hoc Networks" (DSR) [4], the "Optimized Link State Routing Protocol" (OLSR) [5], and the "Topology Broadcast based on Reverse-Path Forwarding" (TBRPF) [6], are being supported; AODV, OLSR and TBRPF have reached the *Request For Comments* (RFC) level.

OLSR and TBRPF are proactive routing protocols that periodically send "Hello" messages for link state sensing. The delay necessary to detect a broken link can be calculated as:  $BLT = HI \times MHL$ , where  $HI$  is the average Hello interval and  $MHL$  is the minimum number of consecutive Hellos lost that triggers a broken link event. Since  $1 \leq HI \leq 2$  seconds and  $2 \leq MHL \leq 3$ , nodes will typically require between 2 to 6 seconds until the network topology updating task is activated. AODV and DSR are reactive routing protocols that typically use information from lower layers in order to detect broken links earlier. However, even though we achieve lower reaction times to link changes by enabling link awareness, the re-routing process can still introduce quite long disconnection periods.

IETF's RFC 3357 [7] defines two metrics, namely the "loss distance" and the "loss period", and the associated statistics that together capture loss patterns experienced by packet streams on the Internet through probes. However, the statistics proposed do not offer an in depth view of the packet loss phenomena.

In this work we extend some of the concepts presented in [7] and we propose new metrics aiming specifically at audio and video streams. Moreover, we propose two models for packet arrivals and packet loss patterns in MANETs based on hidden Markov models [8] (HMMs) theory. Though the models derived could be used in other packet networks, our focus on MANETs is due to the unusually large packet loss bursts that are prone to occur in these networks. We believe that such packet loss patterns impose great demands on the model, so that model validation is done in an extreme situation.

Hidden Markov models have initially been developed to address the requirements of speech recognition. However their use has spread to several other areas, like the computer networks area. Wei, Wang, et al. propose in [9] a solution based on modeling that uses periodic end-to-end probes to identify whether a "dominant congested link" exists along an end-to-end path. In [10] Liu, Matta

and Crovella obtain an improved TCP version through end-to-end differentiation between wireless and congestion losses, providing effective operation in hybrid wired/wireless environments. Their approaches integrate HMMs with packet loss pairs (PLP).

In the literature we can also find uses of the simpler Markov chains to model Internet behaviors, such as [11, 12].

By modeling the packet error bursts in an error prone network environment we achieve interesting benefits in terms of simulation time and required resources for our simulation experiments, keeping a good representation of real-life packet loss bursts.

In section 2 we describe our model and the methodology followed, and we also present the two proposed models for the packet loss burst in multihop wireless paths. Section 3 presents the validation of those models, also defining a set of metrics for packet loss bursts. These metrics allow assessing how different routing protocols perform during a well-defined period in terms of loss bursts. Section 4 illustrates with an application example the simulation time and resources saved when using the models derived. Finally, section 5 presents the conclusions of our work, along with references to future work.

## 2. MODEL DESCRIPTION AND PROPOSED METHODOLOGY

In a previous work [13] we found that routing related losses can provoke quite large packet loss bursts, even when operating in ideal conditions.

We assume that stations belonging to the MANET are found in different routing states (e.g. route available, route discovery, re-routing, etc.). Anyway, independently of the routing state, packet losses can occur for a variety of other reasons (collisions, channel noise, queue dropping, etc.). Therefore, an outside observer cannot relate a packet loss with a certain routing state. We deal with a situation where the observation is a probabilistic function of the state, that is, only the output of the system and not the state transitions are visible to an observer. We will therefore try to solve the classification problem using a *hidden Markov model* (HMM) [8].

HMMs are well known for their effectiveness in modeling bursty behavior, relatively easy configuration, quick execution times and general application. So, we consider that they fit our purpose of accelerating the evaluation of multimedia streaming applications adequately, while offering similar results as with simulation or real-life testbeds.

### 2.1 General methodology

Relatively to the methodology proposed in this work, we start by selecting a single data stream (e.g. audio, video, etc.) for analysis, as well as the criteria for considering a packet good or unusable by the application. We can take into account factors such as which packet arrives to destination within a maximum delay, the delay jitter limits, the dependency among packets, etc. We then have to map each packet sequence number with values 1 - considering the packet good - or 0 if the packet does not arrive to destination or does not meet any of the chosen criteria. This output mapping is stored in a trace file named *ST*, that will be parsed to obtain the distributions of consecutive packets arriving (CPA) and consecutive packets lost (CPL) stored respectively in trace files *CI* and *CO*. We then use the latter two to tune the proposed HMMs.

In a HMM the number of states is not defined by the possible output events. To choose an adequate HMM configuration we propose starting from a very simple 2-state model as presented in the next section. We consider that one of the states models a currently broken path, where the probability for a packet to reach destination

is zero. The other states models path availability, and the probability for a packet to reach destination is given by function  $h(s)$ , where  $s$  is the packet size. This function models packets lost mostly due to collisions, but also due to channel noise, packet fragmentation, buffers overflow, and the type of MAC used.

Starting from the 2-state model, we can compare the model's output with the distributions used for its tuning, and assess if the desired degree of accuracy is achieved. If the results are not accurate, we have to add more states to the model and repeat the process until the results are satisfactory.

The characteristics of the routing protocol used can be useful to provide an insight on how to enhance the model (see section 2.3 for an example). In our experiments we did not have to use more than three states, showing that the model complexity can be kept low and still provide the desired behavior.

In the two following sections we show how to model the transmission of data streams on MANETs using routing protocols such as OLSR or DSR using 2-states and 3-states HMMs. In order to speed up the determination of the optimum values for the model parameters we also present, for each case and for each parameter, a set of heuristics that offer good estimates.

### 2.2 Two-states packet loss burst model

In this section we present the simplest HMM that is able to model large lost bursts. The idea is to focus on two distinct situations: when a path towards the destination is lost and no packet can arrive successfully, and when a path to the destination exists but some of the packets are dropped due to congestion, transmission errors, buffers overflow, etc. It consists of a two-states HMM based on the Markov chain shown in Figure 1 (also known as the Gilbert model).

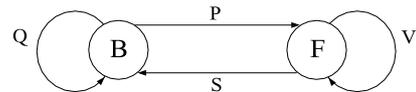


Figure 1: Two-state Markov chain for the multi-hop wireless path model

State B models the situation where a path towards the destination has been lost; the probability for a packet to reach the destination is zero. In state F packets arrive to the destination with a probability defined by function  $h(s)$ , where  $s$  is the packet size. Mapping state B with 0 and state F with 1 we obtain the following transition probability matrix:

$$A_2 = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix} = \begin{bmatrix} Q & P \\ S & V \end{bmatrix} \quad (1)$$

In this work we estimate the different parameters of the HMM using experiments based on the ns-2 simulator [14] as input. We have tested several different scenarios with different mobility and traffic patterns, and we have chosen one that was particularly representative in terms of large packet loss bursts. This choice aimed at stressing the model using a very demanding example.

Our setup consists of a 200 m × 200 m indoor scenario with 80 nodes. The wireless interfaces are based on the IEEE 802.11b standard with radio range limited to 50 m. The medium access used is the distributed coordination function (DCF). The node mobility is generated using the random waypoint model with node speed between 0 and 2.4 m/s. The source of the reference flow sends packets with random sizes ranging between 10 and 2300 bytes at a rate of 50 packets per second. The background traffic consists of 4 UDP sources generating 512 bytes packets at a rate of 4 pkt/s. We evaluate both a reactive (DSR) and a proactive (OLSR) routing protocol. Applying a filter to the simulation's output we obtain

a trace file (*ST*) where incrementing packet sequence numbers are tagged with either a 1 or a 0 (for packets received and packets lost respectively). Our criteria is that all packets that arrive to destination in less than 300 ms are considered good packets (tagged 1); the remaining were tagged as lost (0). From this trace we obtain two other with the lengths of the sequences of *consecutive packets lost* (trace *CO*) and of the sequences of *consecutive packets received* (trace *CI*). These traces will be used as training sequences for the model calculation.

Using trace *ST* we first analyzed the correlation between packets size and the event of losing, or not, a packet. The correlation coefficient found is  $r^2 = 6.03 \times 10^{-6}$ , which indicates that the event of losing a packet is basically independent from packet size. Therefore, the probability function associated to state F will be fixed at a constant value  $h(s) = H$ .

Using trace *CO* we calculate the ratio between the total number of packets lost and the sum of the lengths of CPLs sequences bigger than one. With trace *CI* we do the same with the packets received. We find that the ratios for the packets received is high (above 99%), as expected. The interesting result is that the ratios for packets dropped are also high (above 97%), indicating that packet loss bursts are the dominant cause of losses, contrarily to a random-loss situation. Since the main reason for packets lost in sequence is a route failure, these events shall take place mostly in state B.

From these results, and since parameter H accounts mainly for non-consecutive packet losses, we consider that  $H = 1 - \varepsilon \approx 1$ . This allows us to find the vector of estimated parameter values  $v_e = (S_e, P_e, H_e)$ . We estimate  $S_e$  and  $P_e$  taking into account that the runs at each state of a Markov chain are memoryless, having by definition a geometric distribution. Using this information we find that run lengths for B and F states have an average size of  $\frac{1}{P}$  and  $\frac{1}{S}$  respectively. Therefore

$$S_e = \frac{1}{\mu_c}, \text{ and } P_e = \frac{1}{\mu_b},$$

where  $\mu_c$  is the average length of the sequences of consecutive packets arriving (CPA), and  $\mu_b$  is the average length of the consecutive packets lost (CPL), where  $CPL > 1$ , that is after removing all isolated packet losses.

The value  $H_e$  is estimated using the transition probability matrix  $A_2$ . We can find the steady-state probability  $\pi$  for all states by evaluating  $\pi = \pi A_2$ . After finding  $\pi$  we can define the exact probability for a packet to arrive to destination,  $p_{arrival}$ , using the following expression:

$$p_{arrival} = H_e \cdot \pi_1 = H_e \cdot \frac{P_e}{P_e + S_e} \quad (2)$$

Since we have already estimated values for P and S, and since  $p_{arrival}$  can be found using the simulation results, we can obtain from Equation 2 the value for  $H_e$ .

Starting from the vector of estimated parameter values  $v_e = (S_e, P_e, H_e)$ , we proceeded to find a more precise solution through an iterative process, which can be any one of the many available in the literature [8]. We consider that our estimates  $v_e$  are close to the definitive ones, and so the method we use is a hybrid iterative/brute force technique. Starting from the estimated parameter values we select a search interval for each parameter testing several points in this interval and choosing the one that minimizes error function  $f$ . In the next iteration we reduce the search interval around the point that minimizes  $f$  in the previous iteration. We proceed with this algorithm until the output from function  $f$  is smaller than a pre-defined error value ( $\xi$ ). This value defines the desired degree of accuracy of the model.

The minimization function we used was:

$$f = \left| \frac{\mu_{CPA-M} - \mu_{CPA-S}}{\mu_{CPA-S}} \right| + \left| \frac{\mu_{CPL-M} - \mu_{CPL-S}}{\mu_{CPL-S}} \right| \quad (3)$$

where  $\mu_{CPA-S}$  and  $\mu_{CPA-M}$  refer to the mean values of the consecutive packet arrival distribution for the simulator and the model output respectively, and  $\mu_{CPL-S}$  and  $\mu_{CPL-M}$  refer to the mean values of the consecutive packets lost distributions. We have chosen this function for minimization since it also allows to set bounds on the probability of packet arrivals. If we impose that  $f < \xi$ , and since  $p_{arrival}$  can also be defined as:

$$p_{arrival} = \frac{\mu_{CPA}}{\mu_{CPA} + \mu_{CPL}} \quad (4)$$

we find that the relative error for  $p_{arrival}$  ( $e$ ) is bounded by  $\frac{1-\xi}{1+\xi} < e < \frac{1+\xi}{1-\xi}$ . We consider  $f$  a good choice because similar values for  $p_{arrival}$  obtained from the simulator and the model will allow us to perform consistent comparisons when evaluating multimedia application's software. In fact, if we achieve similar distributions for CPL and CPA but do not achieve very similar values of  $p_{arrival}$ , it would not be possible to validate the model against the simulator correctly. It would mean that different goodput values are achieved with the simulator and with the model, making any kind of comparison unfair.

Table 1 presents both the  $v_e$  values and the values obtained through the iterative process ( $v_i$ ).

**Table 1: Estimated parameters values ( $v_e$ ) vs. the values obtained through the iterative process ( $v_i$ ).**

	DSR		OLSR	
	$v_e$	$v_i$	$v_e$	$v_i$
<i>P</i>	$10.79 \times 10^{-3}$	$11 \times 10^{-3}$	$5.44 \times 10^{-3}$	$5 \times 10^{-3}$
<i>S</i>	$1.36 \times 10^{-3}$	$1.3 \times 10^{-3}$	$2.87 \times 10^{-3}$	$1.85 \times 10^{-3}$
<i>H</i>	0.99560	0.99998	0.969	0.999

In Table 2 we present a comparison, in terms of consecutive packets arriving (CPA) and consecutive packets lost (CPL), using the simulator results, i.e., traces *CI* and *CO*, and the two-states model using either the values of vector  $v_e$  or vector  $v_i$ .

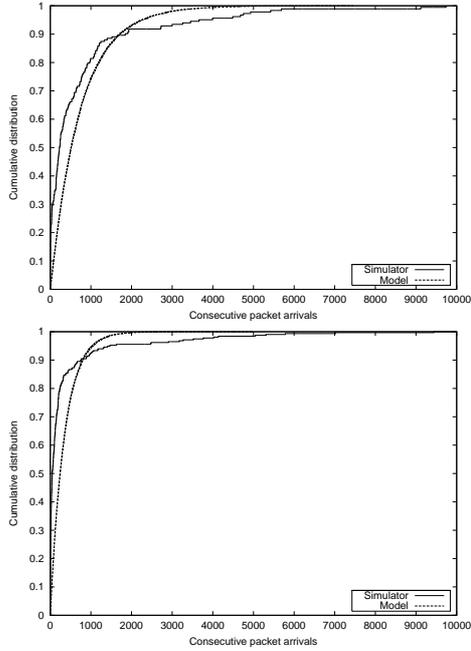
**Table 2: Statistical average matching for the estimated and iterated model values**

DSR	Simulator	Model			
		$v_e$	error	$v_i$	error
$\mu_{CPA}$	737,04	554,65	24,75%	746,58	1,29%
$\mu_{CPL}$	86,91	71,53	17,70%	88,99	2,39%

OLSR	Simulator	Model			
		$v_e$	error	$v_i$	error
$\mu_{CPA}$	348,69	29,58	91,54%	346,96	0,50%
$\mu_{CPL}$	129,99	17,49	86,55%	129,92	0,05%

Figures 2 and 3 show a comparison of the consecutive packet arrivals patterns and the consecutive packet loss patterns respectively. Using vector  $v_i$  we compare the probability density function and the cumulative distribution function for the simulation and for the model outputs. From figure 2 we can observe that the statistical distribution provided by the model has a close resemblance with the simulator output.

Concerning the distribution of consecutive packet losses, figure 3 shows that the two-state model fails in accurately modeling the desired consecutive packets loss pattern for DSR. Concerning OLSR,



**Figure 2: Cumulative distribution function of consecutive packet arrivals (CPA) for DSR (top) and OLSR (bottom).**

we consider that the HMM is able to approximate the consecutive packet loss distribution satisfactory.

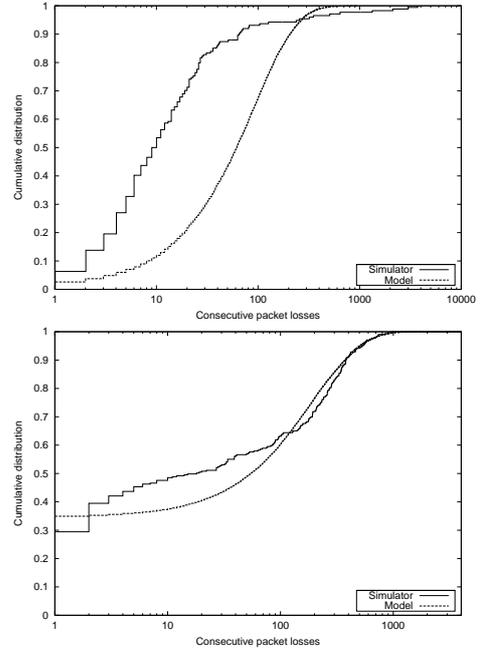
The different precision of the results for the two routing protocols is due to their different routing nature. DSR belongs to the reactive family of protocols. These protocols are able to reestablish a path very quickly until there are no more available routes on the source node's cache. Afterwards they have to proceed with the possibly high time-consuming process of route discovery until communication is resumed. Proactive protocols such as OLSR and TBRPF rely on frequent "Hello" and topology update messages to manage the routing tables. Therefore, these are not prone to present the asymmetry encountered with DSR, being more closely modeled with the two-states HMM presented before. Modeling more accurately DSR's distribution for consecutive packet losses can be done at the cost of introducing more complexity to the model. In the next section we show how a more accurate approach can be described through a three-state Markov model.

### 2.3 Three-states packet loss burst model

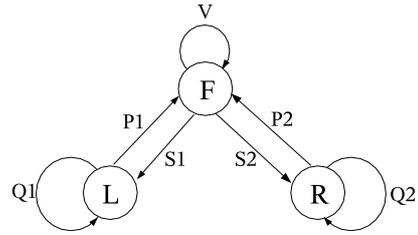
In this section we present an enhancement of the model described in the previous section which obtains a much better approximation of DSR's packet loss bursts distribution. Analyzing DSR's behavior we find that path breaks can be either short if breakage is handled by a quick re-routing process using the node's cache, or long if a route discovery process is required. Taking into account this different behavior, we replace state B from the two-states model with states L and R, where state L models short path breakages and state R models route discovery processes (R). The resulting three-states HMM is shown in Figure 4.

As in the two-states model, packets arrive at destination in state F only, with probability  $H$ . In states L and R all packets are lost. Mapping state L as 0, state F as 1, and state R as 2, we obtain the following transition probability matrix:

$$A_3 = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{bmatrix} = \begin{bmatrix} Q_1 & P_1 & 0 \\ S_1 & V & S_2 \\ 0 & P_2 & Q_2 \end{bmatrix} \quad (5)$$



**Figure 3: Cumulative distribution function of consecutive lost packets (CPL) for DSR (top) and OLSR (bottom).**



**Figure 4: Three-states Markov chain for the multi-hop wireless path model**

We work with the traces described in the previous section: trace  $ST$  with the mapping between packet sequence number and packet received/lost events, trace  $C0$  with the lengths of the sequences of packets lost, and trace  $C1$  with the lengths of the sequences of packets received. These traces will be used as training sequences for the model calculation.

As in the previous section, we maintain  $H = 1 - \varepsilon \approx 1$ . We classify consecutive packets lost (CPL) events into two groups choosing a threshold  $t$ . The value for  $t$  can be chosen by determining the inflection point of the cumulative distribution for consecutive packets lost, or using any other criteria. Notice that the determination of the final parameter values through iteration are independent of the threshold  $t$ , but better guesses for  $t$  allow finding the final values with fewer iterations. In this example, the value we have chosen for  $t$  is 200 (see figure 5), which is slightly above the inflection point and corresponds to 4 seconds at a source rate of 50 pkt/s.

We now wish to determine vector  $v_e = (S_{1_e}, S_{2_e}, P_{1_e}, P_{2_e}, H_e)$  relative to estimated values. We consider that  $\mu_c$  is the average length of the sequences of consecutive packets arriving (CPA),  $\mu_b$  is the average length of the consecutive packets lost (CPL) when their length is greater than 1 and less or equal than  $t - 1$ , and  $\mu_B$  is the average length of the CPL when their length is equal to or greater than  $t$ . We can then calculate the values for  $S_{1_e}$ ,  $S_{2_e}$ ,  $P_{1_e}$  and  $P_{2_e}$  using the following equations:

**Table 3: Estimated parameters values ( $v_e$ ) vs. the values obtained through the iterative process ( $v_i$ ).**

	$v_e$	$v_i$
$S_1$	$1.2735 \times 10^{-3}$	$1.173 \times 10^{-3}$
$S_2$	$0.08324 \times 10^{-3}$	$0.07669 \times 10^{-3}$
$P_1$	$59.21 \times 10^{-3}$	$59.2 \times 10^{-3}$
$P_2$	$0.79821 \times 10^{-3}$	$0.7982 \times 10^{-3}$
$\bar{H}$	0.99916	0.9999

**Table 4: Statistical average matching for the estimated and iterated model values**

DSR	Simulator	Model			
		$v_e$	error	$v_i$	error
$\mu_{CPL}$	86,91	28,59	67,1%	85,82	1,25%
$\mu_{CPA}$	737,04	268,15	63,6%	737,12	0,01%

$$S_{1_e} = \frac{1}{\mu_c} \cdot P(\text{CPL} < t \mid \text{CPL} > 1), \quad (6)$$

$$S_{2_e} = \frac{1}{\mu_c} \cdot P(\text{CPL} \geq t \mid \text{CPL} > 1) \quad (7)$$

$$P_{1_e} = \frac{1}{\mu_b}, \text{ and } P_{2_e} = \frac{1}{\mu_B} \quad (8)$$

The values obtained from these expressions allow to evaluate the transition probability matrix  $A_3$ , and after that to determine the steady-state probability for all states,  $\pi$ , obtaining:

$$p(F) = \pi_1 = \left(1 + \frac{S_{1_e}}{P_{1_e}} + \frac{S_{2_e}}{P_{2_e}}\right)^{-1} \quad (9)$$

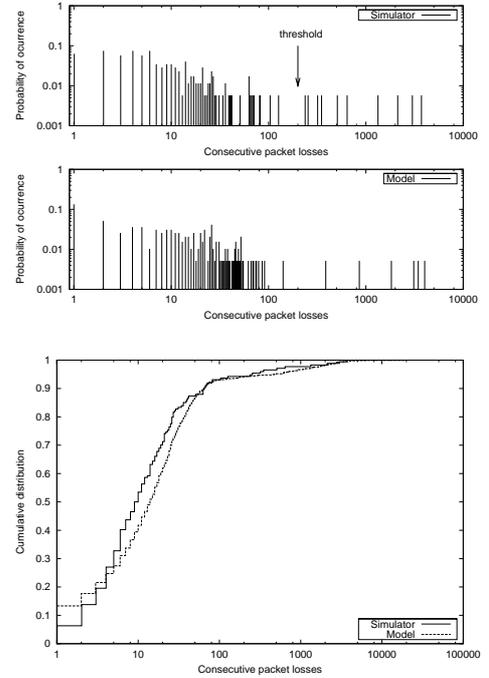
The expression  $p_{arrival} = H_e \cdot \pi_1$  gives us the exact probability that a packet arrives to destination, and it is used to calculate the value for  $H_e$ , thus completely defining vector  $v_e$ . We then find the final parameters values using the same methods exposed in the previous section. Table 3 presents both the vector of estimated values ( $v_e$ ) and the vector of values obtained through the iterative process ( $v_i$ ).

Table 4 presents a comparison of the mean errors when comparing traces  $CI$  and  $CO$  obtained from the simulator with the same traces obtained using the three-states model using either the values from vector  $v_e$  or from vector  $v_i$ . The comparison is made in terms of consecutive packets arriving (CPA) and consecutive packets lost (CPL). We observe that now the probability density function and cumulative distribution function obtained with the model fit the desired distribution with a much higher degree of accuracy, as shown in figure 5. It is evident that the introduction of two loss-states instead of one improves the behavior of the model's cumulative distribution curve. We also observe from the probability density function that our model can reproduce very large bursts.

Concerning the consecutive packet arrivals distribution, both density and cumulative distribution functions are very similar to the ones obtained with the two-states model, as expected. Though the model could be further extended in order to achieve small values of consecutive packet arrivals, thus offering a better approach to the cumulative distribution curve of the simulator, we consider that it is an irrelevant issue to our purpose.

### 3. VALIDATION

We now validate the models proposed in Sections 2.2 and 2.3, verifying their correctness and adequateness for the purpose of evaluating multimedia streaming applications.



**Figure 5: Probability density function and cumulative distribution function for packet loss bursts**

For that purpose, we will define a set of metrics that measure packet loss bursts. Then, we will get these metric values for both the simulator and model outputs in order to verify the effectiveness of our models.

#### 3.1 Measuring packet loss bursts

Before detailing the different metrics proposed to characterize packet loss bursts, we provide a definition of the boundaries of a packet loss burst specifically designed for video and audio data flows. We consider that data flows belonging to different applications will not be affected by packet loss bursts in the same way. It is also important to point out that loss burst measurements are always done focusing on a single traffic flow, and not for all the traffic in the network, even if there are other similar flows.

We define the *burst start threshold* as the minimum number of consecutive packets that have to be lost to consider the presence of a burst. We also define the *burst end threshold*, as the minimum number of consecutive packets arriving to the destination after a loss burst to consider that communication has been adequately resumed. The burst start and burst end thresholds will depend on the type of information sent and the packetization granularity. For example, if we consider that at least one entire video frame has to be lost for a burst to be meaningful, and that one entire frame has to arrive for communication to be resumed, the burst start threshold and the burst end threshold will have the same value and it will be equal to the number of packets per frame defined in the video codec.

We will now proceed with the definition of some indicators to describe packet loss burst occurrences. The most simple indicator is *packet burst percentage* (PBP), defined as:

$$PBP(\%) = \frac{\sum_{i=1}^K B_i}{N} \quad (10)$$

where  $B_i$  is the size of loss burst  $i$  in number of packets,  $K$  is the total number of loss bursts and  $N$  the total number of packets sent. The PBP gives a measure of the relative burst incidence. To mea-

sure the relative impact of bursts over the total number of packets lost  $L$ , we define the *Relative Burstiness* (RB) metric as:

$$RB = \frac{\sum_{i=1}^K B_i}{L}, 0 \leq RB \leq 1 \quad (11)$$

In a situation where most packets are lost in a random manner the RB parameter approaches 0, while when packet loss bursts dominate, RB will be greater than 0.5.

This parameter allows us to detect where the network needs more improvements: if on the routing protocol side ( $RB > 0.5$ ) or on the MAC support for traffic flows ( $RB < 0.5$ ).

Both these indicators are burst size independent. They equally penalize very small bursts occurring in a distributed fashion and very large bursts if the total number of packets lost is the same. From the user point of view, however, long communication breaks may be unacceptable. To take into account such discrepancies, we introduce the *Burstiness Factor* (BF):

$$BF = \frac{\sqrt{\sum_{i=1}^K B_i^2}}{N}, 0 \leq BF \leq 1 \quad (12)$$

The BF is a metric of the impact of the re-routing time of different routing protocols over a given flow; smaller values indicate that interruptions caused by routing protocols are either fewer or smaller.

Though BF is a good indicator to measure improvements on routing protocols, it does not take into account the relative position of the bursts, which can have different impact on multimedia streams from a codec point of view. We therefore introduce the *media smoothness factor* (MSF):

$$MSF = \frac{\sqrt{\sum_{i=1}^T F_i^2}}{N}, 0 \leq MSF \leq 1 \quad (13)$$

where  $T$  is the total number of *inter-bursts* or *burst delimited* periods, identified as  $F_i$ , and  $N$  is the total number of packets. Figure 6 shows an example of plotting the values of  $F_i$  and  $B_i$ . In this example the burst start and burst end thresholds are set to 3 packets, thus obtaining  $K = 2$  and  $T = 3$ . Applying this threshold we have two well defined loss bursts and three burst delimited zones.

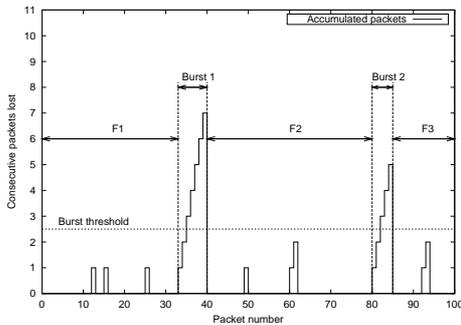


Figure 6: Example of plotting the values of  $F_i$  and  $B_i$

The MSF measures the *fluidity* experienced by a multimedia data stream; obviously,  $MSF \gg BF$  must hold for communication to be sustainable. To better understand the different properties of BF and MSF, we propose a case study scenario, depicted in figure 7, where we have a train of  $K$  bursts of length  $G$ , separated by exactly  $X$  packets. The burst sequence is centered so that  $Y$  packets separate the first and last bursts from the beginning and end of the observation period, where  $Y = (N - K \cdot G - (K - 1) \cdot X)/2$ .

In this scenario  $BF = \sqrt{K} \times G/N$ , which is independent from

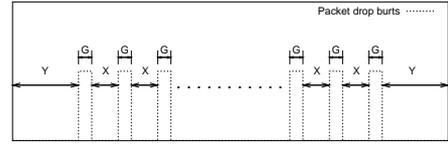


Figure 7: The BF vs MSF case study scenario

the bursts separation value ( $X$ ), while:

$$MSF = \frac{\sqrt{(K-1) \cdot X^2 + 2 \cdot Y^2}}{N}, \quad (14)$$

depends not only on the size and number of bursts, but also on the distance between them. Considering that the upper limit for  $X$  when  $Y = 0$  is given by:

$$X_{max} = \frac{N - K \cdot G}{K - 1} \quad (15)$$

we can normalize Equation 14 using  $z = X/X_{max}$ , obtaining:

$$MSF = \frac{N - K \cdot G}{N} \cdot \sqrt{\frac{z^2}{K-1} + \frac{1}{2} \cdot (1-z)^2} \quad (16)$$

Figure 8 shows the behavior of Equation 16 as a function of  $K$ , taking  $\frac{G}{N} = 0.02$ .

Equation 14 reaches its minimum when  $x_m = y_m = \frac{N-K \cdot G}{K+1}$ . This indicates that the minimum value of MSF is reached when interruptions on communication are evenly separated, that is, when distance between loss bursts is equal to the distance to the extremes.

The normalized expression for  $z_{min}$  is:

$$z_{min} = \frac{x_m}{X_{max}} = \frac{K-1}{K+1}, K \geq 2 \quad (17)$$

which depends solely on the number of loss bursts present on the sequence.

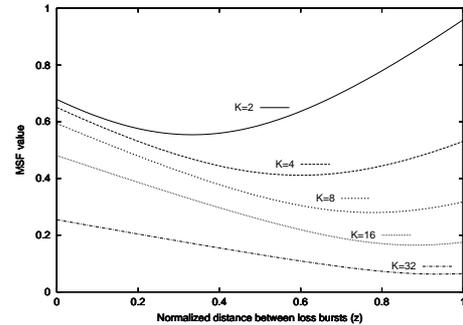


Figure 8: MSF variation with distance between bursts

We can directly check the results from figure 8, and also check that it approaches 1 for large values of  $K$ . Since typically we will have a large number of gaps ( $K \gg 1$ ), the MSF will be monotonically decreasing. This result allows us to conclude that MSF offers a measure of burst concentration for similar values of BF, increasing as the concentration of bursts increases.

Finally, we have defined four metrics for analyzing packet loss bursts: the PBP (burst percentage), the RB (relative burstiness), the BF (burstiness factor), and the MSF (media smoothness factor). These metrics give us different information about loss burst patterns, and they will help us in the model validation process.

### 3.2 Validation process

We now apply the previously defined metrics to compare the two-states and three-states HMMs results with the simulator's output when using the DSR protocol.

We set the burst start threshold equal to the burst end threshold for the sake of simplicity in the presentation of results.

The bursts percentage over the total number of packets sent (PBP) and the RB parameter vary with increasing thresholds for burst start/end values, see figure 9.

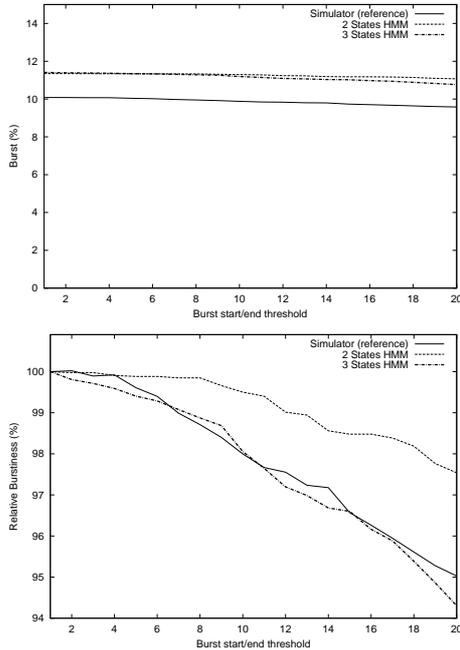


Figure 9: PBP (top) and RB (bottom) comparison

The RB parameter represents clearly the relation between packet losses that pertain to bursts and those that don't. As it can be seen, the three-states model approaches the reference curve from the simulator with greater accuracy than the two-states model.

In figure 10 we compare the output from the HMMs and the simulator in terms of the Burstiness Factor (BF) and the Media Smoothness Factor (MSF).

We observe that the results for the three-states HMM are much closer to the reference values. We consider that the degree of accuracy achieved is acceptable for applications such as video codec enhancing and tuning. In terms of the MSF, which takes into account consecutive packet arrivals instead, we observe that the three-states HMM approaches the reference MSF value with increasing thresholds. The slight difference is expected since the accuracy of the consecutive packet arrivals distribution was not the main focus of our model. It could be improved by increasing the number of states in the HMM, similarly to was done for DSR's consecutive packet losses distribution.

To further validate our model, we now proceed comparing the results in terms of video quality achieved with the model against the ones obtained with the simulator. To perform this comparison we replace the randomly generated data stream used initially with streams of well-known QCIF video sequences, namely: *foreman*, *container* and *flower*, each replicated to be 300 seconds long. In our analysis we used the H.264 video codec framework [15] in order to obtain both the input video trace files and the output video quality results for the simulator and the model.

The metric we use is video distortion, also known as Peak Signal-to-Noise Ratio (PSNR), which is the most commonly used objective video quality metric.

In figure 11 we present a box-plot comparison between the simulation values and the model values using the three video sequences

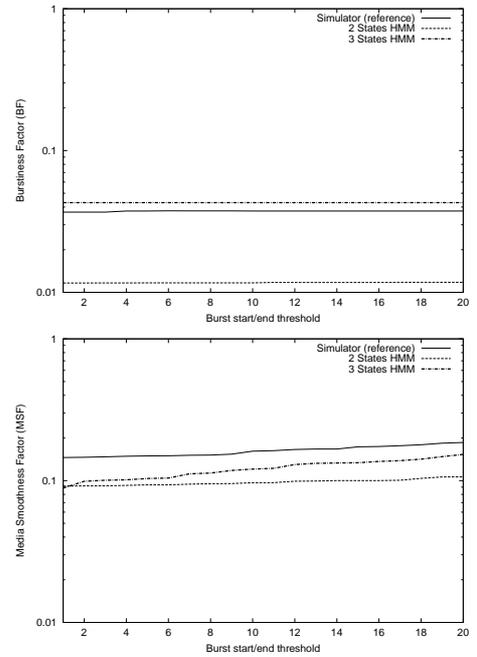


Figure 10: BF (top) and MSF (bottom) comparison

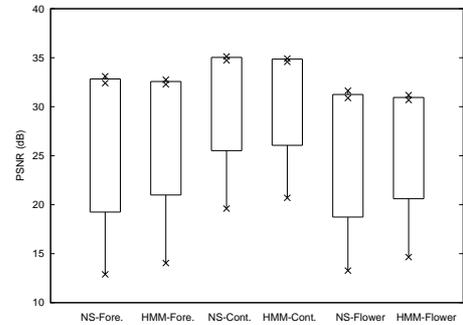


Figure 11: Box plots for the video distortion achieved with the simulator and HMMs for the *foreman*, *container* and *flower* QCIF video sequences.

under test. The box plot represents the minimum, the median, and the maximum values (the three crosses), and the box contains the values between the 0.250 and the 0.750 quantiles of the data. The figure clearly shows that the results achieved with the model closely resemble the ones achieved via simulation. These results, along with the previous ones, allow us to conclude that the model proposed offers a behavior quite similar to the one we wish to obtain.

#### 4. A MODEL'S APPLICATION EXAMPLE

In this section, to illustrate the applicability of our models, we use them as a tool to speed up the evaluation and tuning of a video codec (in this example, an H.264 video codec). We measure the impact of the different steps required for simulation and data extraction using the ns-2 simulator and HMMs for both DSR and OLSR. We simulate the streaming of a typical movie 1 hour and 30 minutes long. Results presented in table 5 allow comparing the time consumed at each step using the ns-2 simulator only or using HMMs when the chosen routing protocol is DSR. The values presented are achieved on a dual 2,6 GHz Pentium-IV server with 2 Gbytes of RAM running GNU/Linux version 2.4.22.

Notice that when using only ns-2 steps I, II and III must be re-

**Table 5: Duration of different simulation steps using the ns-2 simulator or HMMs**

	ns-2	HMMs
I- Mobility generation (s)	840	840
II- Single simulation time (s)	1320	1320
III- Extraction of packet loss details (s)	60	60
IV- Determining model parameters (s)	-	3600
V- Single simulation time using HMM (s)	-	0,40
100 simulations total time	61h40m	1h38m

peated every time. When using HMMs steps I to IV are only performed once, and step V is the only one repeated.

Results show that almost all the time is consumed in simulation and in the determination of model parameters. Once that is done, though, the execution of the model is very quick. Relatively to the entry named “Determination of model parameters”, we wish to point out that this time takes into account not only the time to determine the initial estimates for the different parameters ( $v_e$ ), but also to find the final iterated values ( $v_i$ ). In the bottom of both tables we present the estimated time to run 100 simulations, a value required to extract statistically significant results.

Relatively to the improvements achieved, we find out that our algorithm allows execution 38 times faster when using DSR, and up to 74 times faster when using OLSR. This difference is due to the time taken by each simulation run when using OLSR: 9720 s.

In terms of trace file output we find that, comparing trace file sizes, the model’s output is 300 to 12000 times smaller than the simulator’s output, though the output from the last can be reduced. Concerning real-life experiments, the trace file size can be reduced to the size of the HMM’s trace file.

Optimal tuning of the video codec using both the simulator’s output and the HMM’s output was also performed. We find that the most error-resilient parameter choices for the codec are the same with both solutions, which allows us to conclude that the methods and techniques exposed in this paper fit our purpose adequately.

## 5. CONCLUSIONS

In this work we proposed an alternative to evaluate multimedia streaming applications in MANETs avoiding repetitive, time-consuming simulations. Our solution, based on hidden Markov models, allows to evaluate the effects of packet loss and arrival patterns when streaming a compressed audio/video sequence in MANETs using different routing protocols.

Results show that a two-states model is effective in modeling packet loss bursts when using a proactive protocol such as OLSR, though failing to accurately model MANET behavior when using a reactive protocol such as DSR. This occurs because DSR’s mechanisms present a higher level of asymmetry, thus requiring a three-states HMM.

We finally validated our models showing that the proposed HMMs provide similar results in terms of the loss burst metrics we defined and also in terms of video distortion.

We showed an application of our models to the design and evaluation of a video codec. We obtain very significant gains in terms of simulation time and disk space usage, while achieving the same configuration in terms of optimal codec tuning.

Overall, we consider that the strategy presented in this paper has proved to be an adequate alternative to the developers of multimedia streaming applications for MANETs, showing excellent results in terms of both accuracy achieved and speedup.

As future work we plan to validate both models in real-life scenarios using both families of routing protocols and different traffic/mobility patterns.

## 6. REFERENCES

- [1] Internet Engineering Task Force. Manet working group charter. <http://www.ietf.org/html.charters/manet-charter.html>.
- [2] International Standard for Information Technology - Telecom. and Information exchange between systems - Local and Metropolitan Area Networks - Specific Requirements - Part 11: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE 802.11 WG, ISO/IEC 8802-11:1999(E) IEEE Std. 802.11, 1999.
- [3] Charles E. Perkins, Elizabeth M. Belding-Royer, and Samir R. Das. Ad hoc on-demand distance vector (AODV) routing. Request for Comments 3561, MANET Working Group, <http://www.ietf.org/rfc/rfc3561.txt>, July 2003. Work in progress.
- [4] David B. Johnson, David A. Maltz, Yih-Chun Hu, and Jorjeta G. Jetcheva. The dynamic source routing protocol for mobile ad hoc networks. Internet Draft, MANET Working Group, draft-ietf-manet-dsr-07.txt, February 2002. Work in progress.
- [5] T. Clausen and P. Jacquet. Optimized link state routing protocol (OLSR). Request for Comments 3626, MANET Working Group, <http://www.ietf.org/rfc/rfc3626.txt>, October 2003. Work in progress.
- [6] R. Ogier, F. Templin, and M. Lewis. Topology dissemination based on reverse-path forwarding (TBRPF). Request for Comments 3684, MANET Working Group, <http://www.ietf.org/rfc/rfc3684.txt>, February 2004. Work in progress.
- [7] R. Koodli and R. Ravikanth. One-way Loss Pattern Sample Metrics. IETF RFC 3357, August 2002.
- [8] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.
- [9] Wei Wei, Bing Wang, Don Towsley, and Jim Kurose. Model-based Identification of Dominant Congested Links. Internet Measurement Workshop Proceedings of the conference on Internet measurement (IMC’03), Miami Beach, Florida, October 2003.
- [10] J. Liu, I. Matta, and M. Crovella. End-to-End Inference of Loss Nature in a Hybrid Wired/Wireless Environment. in Proceedings of Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt’03), Sophia-Antipolis, France, March 2003.
- [11] W. Jiang and H. Schulzrinne. Modeling of packet loss and delay and their effect on real-time multimedia service quality. In *Proc. NOSSDAV*, Chapel Hill, North Carolina, USA, June 2000.
- [12] H. Sanneck, G. Carle, and R. Koodli. A framework model for packet loss metrics based on loss runlengths. In *Proceedings of the SPIE/ACM SIGMM Multimedia Computing and Networking Conference 2000 (MMCN 2000)*, pages 177–187, San Jose, CA, January 2000.
- [13] Carlos T. Calafate, M. P. Malumbres, and Pietro Manzoni. Performance of H.264 compressed video streams over 802.11b based MANETs. In *International Conference on Distributed Computing Systems Workshops (ICDCSW’04)*, Hachioji - Tokyo, Japan, March 2004.
- [14] K. Fall and K. Varadhan. ns notes and documents. The VINT Project. UC Berkeley, LBL, USC/ISI, and Xerox PARC, February 2000.
- [15] Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264 — ISO/IEC 14496-10 AVC), March 2003.