Modeling Multicast Packet Losses in Wireless LANs

Chiping Tang and Philip K. McKinley

Software Engineering and Network Systems Laboratory
Department of Computer Science and Engineering
Michigan State University
East Lansing, Michigan 48824
{tangchip,mckinley}@cse.msu.edu

***Abstract*—**
The characteristics of wireless packet loss have been studied extensively. Most efforts focus on the temporal loss characteristics at individual stations. For multicast protocols and applications, however, the relationship among losses at multiple nodes can directly affect performance. In existing models, packet losses are generally assumed to be spatially independent. In our experiments with an IEEE 802.11 wireless LAN, however, we found that the losses at multiple nodes can exhibit a certain degree of correlation. In this paper, we attempt to quantify this correlation, and demonstrate its effect on multicast communication protocols. Analysis and simulation results show that conventional packet loss models do not adequately capture the loss characteristics exhibited in experimental traces. Therefore, we propose a new approach for modeling packet losses that explicitly accounts for spatial loss correlation. The improved accuracy of the new approach, compared to conventional models, is demonstrated by comparing results of simulations and experiments.

I. Introduction

Group-oriented network applications, such as multimedia conferencing, military command and control, and video streaming of live events, are typically constructed using underlying multicast protocols. In a multicast protocol, a sender transmits packets to multiple receivers simultaneously. Some multicast protocols, such as those used to tranmit files, require strict reliable delivery of data [1], whereas others, such as those used for audio/video streaming [2], are able to tolerate limited packet loss due to the characteristics of the data stream. Error control in multicast protocols is complicated by the fact that a particular packet might be received by some receivers but lost at others, due to disparate environmental conditions.

To predict the performance of multicast protocols through simulation requires accurate models of packet loss on the constituent networks. Unlike models for packet loss in unicast communication, which focus primarily on *temporal* packet loss behavior, models for multicast communication must also consider *spatial* correlation in packet loss among the receiving nodes. As one might expect, these models depend on the type of network. For example, research results show that there exists some degree of spatial packet loss correlation in wide area networks, such as the MBONE [3]. An intuitive explanation is, if a packet is lost somewhere in a multicast tree, all nodes in the downstream branches will experience this loss. On the other hand, most wired local area networks do not exhibit this "downstream effect," since receivers are not organized hierarchically. As a result, the corresponding packet loss models assume losses of multicast packets are spatially independent.

In the past few years, wireless local area networks (WLANs) have become an important component of the Internet infrastructure, and their deployment is continuing at a rapid pace. In addition to supporting unicast connections for individual mobile users, their multicast capability make WLANs well suited to supporting group-oriented applications. Not surprisingly, however, the packet loss characteristics of WLANs differ greatly from their wired counterparts and require a totally different packet loss model. Of particular interest to those researchers studying the behavior of multicast protocols, is the degree of spatial correlation in packet loss for multicast packets.

Wireless packet losses depend on errors at the signal level, which have been studied extensively [4, 5]. It is commonly accepted that two major factors contribute to errors on wireless channels: the deterministic signal energy degradation, and the random noise and fading. Since both depend on the physical relationship between the receiver and the sender (often an access point), wireless channel errors (and hence the packet losses) observed by different receivers are assumed to be spatially independent. Such losses can be simulated efficiently using finite-state Markov models [6]. For example, the two-state Markov model describes wireless channels to be alternatively in either "good" or "bad" states. In good states, few if any packets will be lost during transmission, while in bad states, many packet losses will occur. This model offers a reasonable approximation of wireless packet losses and is widely used due in part to its relatively low computation overhead. More recently, higher order Markov models [7,8] have been proposed to trade computation overhead with model accuracy. at each We refer to such models as the *independent models*.

To investigate the accuracy of such models, we conducted a large number of experiments in a IEEE 802.11 WLAN testbed.

This work was supported in part by the U.S. Department of the Navy, Office of Naval Research under Grant No. N00014-01-1-0744. This work was also supported in part by National Science Foundation grants CDA-9617310, NCR-9706285, CCR-9912407, EIA-0000433 and EIA-0130724.

Somewhat surprisingly, the resultant traces of multicast packet loss clearly exhibit a degree of spatial correlation, even when we factor out losses due to congestion at the access point. We attribute the resulting propagation loss correlation to two phenomena. First, receivers close to a common obstruction or noise source may experience common packet losses. Second, we conjecture that the transmission power at the sender may vary slightly for different packets. Therefore, two receivers at approximately the same distance from the sender are both likely to lose a "weak" packet simultaneously.

We quantify the degree of spatial correlation using a metric, *loss density*, which is the fraction of receivers experiencing a particular packet loss. Both simulation and experimental results show the value of loss density has significant impact on the performance of multicast protocols. Based on these results, we argue that a realistic multicast packet loss model should account for spatial correlation. Although approaches have been proposed to model spatial loss correlation at the signal level [9,10], these methods are computationally intensive and hard to integrate with existing packet-level loss models used in network simulation.

In this paper we propose an efficient multi-stage modeling strategy to account for loss density in wireless multicasting. In Section 2, we discuss prior work in modeling wireless losses and multicast losses, respectively. We also formally define the concept of loss density and demonstrate its effect on multicast protocols. In Section 3, we introduce our experimental environment and our trace collection procedure. We analyze the packet traces and quantify the loss correlation in Section 4. In Section 5, we describe the approach for modeling correlated packet losses, and compare the results with that of the independent models as well as experimental traces. We conclude the paper in Section 6, summarizing the problem and our solution, and pointing out possible future research directions.

II. Background

In this section we present the background of wireless packet loss modeling as well as the related work on spatial loss correlation. We also define the concept of loss density and demonstrate its effect on multicast protocols.

A. Wireless Packet Loss Modeling

Experimental packet traces reveal that packet losses are bursty in wireless networks [7, 11]. In a loss burst, most of the transmitted packets will be lost, while in a loss-free burst, there are few or no packet losses. It is straightforward to use a two-state Markov chain to model the behavior of a wireless channel. The Gilbert-Elliot model [12, 13] is well known and widely used in the research community.

In this model, each channel maintains a channel status flag, which can be either "good" or "bad". While the channel is in the good state, there is very few packet loss; while in the bad state, most packets are lost. The model has several parameters: temporal error correlation, probabilities of a good or bad channel, and the probabilities of error given that the channel is in the good or the bad state. To mitigate the computational complexity, a simplified version of the Gilbert-Elliot model is also in use. In this version, if a packet loss occurs while the channel is in the good state, the channel immediately switches to the bad state. Similarly, whenever a packet is received successfully while the system is in the bad state, the channel switches to the good state. Let us assume the probabilities of staying in the good state and the bad state are α and β, respectively. The probability of any packet loss occurring in the good state is $(1-\alpha)$, and the probability of any successful packet delivery in the bad state is $(1-\beta)$. The model is depicted in Figure 1. We refer the Gilbert-Elliot model to this simplified version in this work.

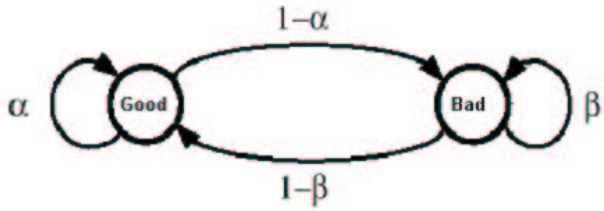

Fig. 1. The simplified Gilbert-Elliot model.

In this model, packet losses are temporally independent. Therefore, both the loss burst and loss-free burst lengths are geometrically distributed. As a result, the good-to-bad transition probability $(1-\alpha)$ is set to the reciprocal of the average loss-free burst length, as measured in a real network, and the bad-to-good transition probability $(1-\beta)$ is set to the reciprocal of the average loss burst length [14]. When generating a trace from the model, the length of each (good or bad) burst is given by

$$L = \frac{log(u)}{log(1-p)},$$

where u is a uniformly distributed random variable with values between 0 and 1, p is the good-to-bad or bad-to-good transition probability, depending on whether it is the loss burst or loss-free burst that is under calculation.

The Gilbert-Elliot model is simple to implement. However, the distributions of loss or loss-free burst length are usually more complicated in real traces. In particular, it is difficult to use one distribution to model the bursts. In [15], Nguyen *et al.* collected a large number of packet traces in a 2Mbps Lucent WaveLAN network. Based on the traces, the burst length distribution is split into several segments. In each segment the distribution is set to a modified exponential or Pareto function with parameters set to best match the real trace in that segment. The values of the parameters are obtained by applying linear regression. The authors showed that this composite model is more realistic than the simple Gilbert-Elliot model in terms of cumulative distribution function of the burst length distribution. An alternative approach is to use higher order Markov process to model packet losses [7, 8]. This method is usually more accurate than the two-state model at the expense of higher computational overhead. Since our focus is on loss density and not on the model for losses at individual receivers, we adopt the simplified Gilbert-Elliot model in this work as the individual loss model for wireless receivers. However, our basic approach could be used in combination with other individual loss models.

B. Spatial Loss Correlation

The multicast packet loss correlation in *wide area* networks has been addressed in a number of research works. In [3], Yajnick *et al.* collected packet traces at multiple sites in the MBONE multicast network. They calculated the loss covariance in each pair of traces and observed certain degree of spatial loss correlation among multicast sites. In [16], the impact of loss correlation is addressed. The authors studied group loss probabilities in shared loss multicast communications for the design of forward error correction algorithms. They presented the cumulative distribution function of successful delivery in a multicast tree, as well as the expected value of packet loss burst length and other properties.

For *local* area networks, however, spatial loss correlation has not been studied extensively. One reason is that the packet loss rate is extremely low in *wired* LANs, while the demand for multicast applications in relatively high-loss *wireless* LANs is still relatively small. With the advent of mobile and ubiquitous computing era, this situation is likely to be changed. Collaborative and interactive learning in the classroom, with temporary wireless LANs comprising students' laptop computers, is one example of wireless multicast applications. Other examples include management of factories and military installations, communication aboard ships, temporary disaster relief sites, and so forth.

The spatial correlation of wireless losses has been well studied at the signal propagation level. It is discovered that the signal strength at multiple antennas is correlated if the antennas are close to each other [10, 17]. Since the wireless signal strength is modeled as random variables, and filtering an uncorrelated random process usually leads to a correlated random process, the modeled signal strength array with a desired covariance matrix can be generated by multiplying the uncorrelated process with any square root of this matrix [18]. Different from these works, we focus on modeling spatial correlation at the packet level in this study. Moreover, we noticed there exists loss correlation even between stations that are well separated.

C. Effects of Loss Density

To quantify the significance of the spatial loss correlation, we introduce the concept of *loss density*, which defines the percentage of the receivers that lose a particular packet. Formally, let N be the total number of receivers, and $n(k)$ the number of receivers that lose packet k. Then the loss density for packet k is, simply:

$$LD(k) = \frac{n(k)}{N}.$$

Loss density is a function of packet number and hence of time. Its value reflects the overall correlation of packet losses among all multicast group members at a particular point in time. Loss density greatly affects the performance of multicast protocols. For example, in NACK-based reliable multicast protocols [1], each receiver sends retransmission request to the sender when a packet loss is detected. The number of receivers sending retransmission request, or equivalently the loss density, will determine the intensity of this feedback traffic. Since the feedback shares the communication channel with the forward data traffic in wireless LANs, the intensity of the feedback will affect the throughput of the reliable multicast protocol. Moreover, in the IEEE 802.11 infrastructure mode [19], the uplink feedback traffic will cause queueing losses at the Access Point (AP) if the sender is located on a connected wired network [20].

In Figure 2, we plot the simulation results of data packet service time (the time needed to acquire the channel and transmit a packet) at the AP versus the number of feedback-sending receivers. In the simulation, the packet size is set to 1400 bytes and the receivers send aggregated feedback for every group of 20 packets. The figure clearly shows the data packet service time will increase as more receivers lose packets and send feedback simultaneously. The slowdown of the data traffic is caused by the increasing degree of channel contention imposed by the feedback traffic at the AP. Clearly, high loss density can dramatically affect communication performance.

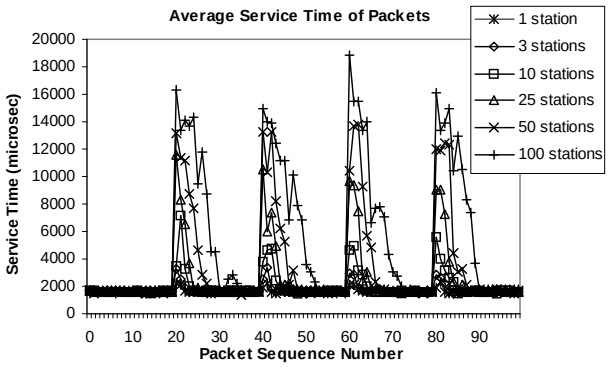

Fig. 2. Effect of feedback traffic on AP service time.

If the sender is located on a wired network, as in many proxy systems [21, 22], the increased service time can cause not only slowdown, but also queueing losses at the AP. If the sender is slow to adjust the data rate, the incoming data packets will soon use up all the buffer space at the AP. We observed this phenomenon in our experiments, in which we configured a wired sender to multicast packets to six wireless receivers. We divide the transmission into nine continuous stages, each of which is approximately 5 minutes long. In statges 1, 3, 5, 7, and 9, none of the receivers sends feedback to the sender. In stage 2, one receiver sends feedback by unicasting. In stage 4, one receiver sends feedback, but using multicast instead of unicast. In stages 6 and 8, five receivers send feedback by unicasting and multicasting, respectively. We plot the average loss rate in each stage in Figure 3, which shows that packet loss is exacerbated when more receivers start sending feedback packets. Moreover, multicasting feedback causes more data packet losses than unicasting does [20]. These results indicate that the loss density, which directly affect the amount of feedback from receivers, has significant impact on the performance of reliable multicast protocols in wireless LANs.

III. Data Collection

In this section we describe the data collection procedure used in this study. Our experimental environment, depicted in Figure 4, consists of a wired LAN and a wireless LAN. The wired

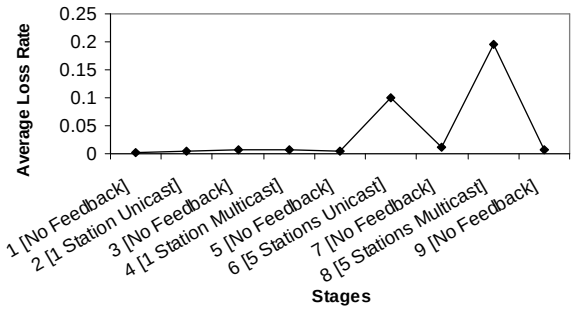

Fig. 3. Effect of feedback traffic on packet loss.

LAN is a 100Mbps Fast Ethernet that connects several workstations. The wired LAN is extended by a Cisco Aironet wireless LAN through a Cisco Aironet 340 Series Base Station. The wireless stations include desktop computers and Dell laptop computers, each configured with a Cisco Aironet 340 or 350 Series Wireless Card. The Aironet network is IEEE 802.11b compatible, using CSMA/CA access control and operating at 2.4GHz. The maximum raw bit rate is 11Mbps, although the achievable thoughput is much lower in practice [23].

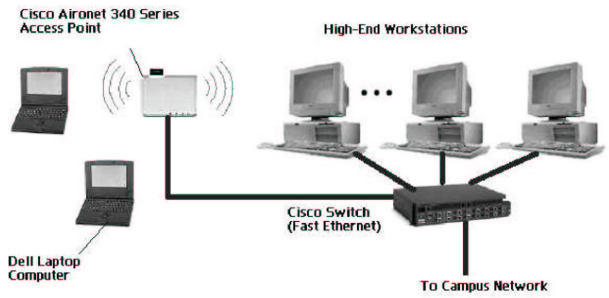

Fig. 4. Experimental environment.

The workstations are mainly Dell desktop computers with 1GHz Pentium III CPU and 512MB memory. The laptop computers have either a 1GHz Pentium III CPU or a 300MHz CPU, and 256MB memory. Both the desktop and laptop computers are configured with either Windows NT or Windows 2000. We built two programs using the Windows socket interface: a sender program and a receiver program. In the experiments we configured the sender process on a wired workstation, while the receivers were located on laptop computers or on desktop computers with wireless network interface cards. This is a typical scenario for content distribution applications involving wireless stations, where the sender might be a proxy redirect the multicast flow from other remote senders.

We conducted a series of experiments and collected packet traces. The sender program transmits data packets at a specified rate to the set of receivers by multicasting. Each data packet is assigned a unique sequence number. The receiver program checks the sequence number of the received packets and records the sequence numbers of lost packets. The program saves these data into packet trace files together with information on corresponding loss burst lengths. From a saved trace we can calculate the average loss burst length, average loss-free burst length, standard deviations and burst length distributions. These values reflect the basic characteristics of packet loss patterns.

Packet losses in a wireless LAN can be caused by reasons other than wireless propagation errors, for example, congestion. Congestion can happen at the access point when the data rate of the wired network is greater than the bandwidth of the wireless network, and at receiver when the receiver program is not running fast enough to process packets and empty kernel buffers before new packets arrive. We have observed very high packet losses when packets are transmitted without flow control [20]. Such losses are very likely caused by buffer overflow. Since we are interested in modeling propagation losses, it is important to exclude these "non-propagation" and protocol-dependent losses from the trace files.

Applying flow control is an effective approach to reduce congestion and queueing losses. Taking into account the theoretical saturation throughput of approximately 7Mbps in an IEEE 802.11b network [23], we use flow control at the sender to limit sending rate at 6Mbps. In case no other traffic exists on the wireless channel, this action will greatly reduce the packet losses caused by congestion. We also collected packet traces at much lower sending rates, such as 2Mbps. At such sending rate, our receiver programs are always able to keep up with data arrivals.

In the experiment, we set up ten wireless receivers, including laptops and wireless card installed desktop computers. They were scattered around the AP, from within 2 meters to about 30 meters away. The configuration and location of the receivers are shown in Figure 5 and Figure 6. Between the two rooms and between the room and the corridor are concrete walls. There are partitions in each room that separate the receivers. We collected 16 packet traces, each 15 minutes long. The packet size is 1400 bytes. The sending rate was set to either 6Mbps or 2Mbps, and there was no other traffic in the wireless LAN.

As mentioned earlier, a critical prerequisite in developing a propagation loss model is to remove any queueing losses from packet traces. Although we inserted inter-packet delays at the sender to limit the sending rate below 6Mbps, and there was no other traffic in the wireless channel, we cannot guarantee that queueing loss does not sometimes occur. One possible cause of queueing loss is the irregular time-consuming operations taking place at the access point, such as the transmission and processing of management packets [19]. Since our focus is to model the wireless multicast propagation losses, and since the amount of queueing losses in packet traces will affect the distribution of loss density, we need to exclude all queueing losses to generate an accurate model. A queueing loss will produce high loss density value (in fact, it will equal 1 for any packets lost). That is to say, a queueing loss at the access point will lead to a packet missing at every receiver. Therefore, only a packet loss shared by all receivers could be a queueing loss. We use this property to guarantee that queueing losses are eliminated from the packet traces.

Let us assume packet losses are spatially independent. We can easily show how likely it is that a propagation loss common to all nodes will occur. Let n be the number of receivers, p_i be the packet loss rate at receiver $i, 0 \leq i < n$. The probability that a common propagation loss occurs is $p = \prod_i p_i$. (Plugging

	node1	node2	node3	node4	node5	node6	node7	node8	node9	node10
Type	D	L	D	D	L	L	L	D	L	L
CPU(MHz)	2*400	1000	1000	1000	300	1000	1000	2*800	1000	1000
Memory(MB)	256	256	512	512	256	256	256	256	256	256
Distance(m)	2	2	4	5	5	10	10	10	20	30

D: Desktop, L:Laptop

Fig. 5. Receiver configurations.

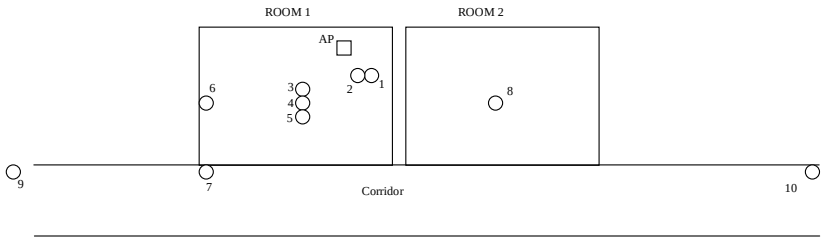

Fig. 6. Receiver locations.

in the average loss rates from our traces, as shown in the next section, and setting $n = 10$, we obtain the probability to be almost 0.)

The occurrence probability of a common propagation loss is extremely low in a typical multicast wireless LAN environment (where $n > 5, p_0 < 0.1$). Consequently, it seems safe for us to take all occurrences of such common losses as queueing losses and exclude them from the packet traces. In reality, however, packet losses at multiple receivers are not independent. A packet in propagation with weak signal power will be probably lost at all receivers. On the other hand, the average loss rate does not reflect the actual loss probability at a particular time point due to the burstiness of wireless losses. Therefore, using this approach to exclude queuing losses might also delete quite a few propagation losses. To estimate the degree of this overkill, we calculate the distribution of loss density in the original traces. Since all traces exhibit similar loss density distribution, we plot only one of them in Figure 7. The figure shows the probability of a loss common to all nodes in the trace is significantly higher than it should be if it were a propagation loss, given the trend in loss density at 0.7, 0.8, 0.9 and 1. On the other hand, the figure also shows that the percentage of losses common to most or all nodes is very small, approximately 0.0001. Therefore, even if we exclude *all* the common losses from the traces, the burst length distribution in the traces will not be significantly affected, while such processing of the traces will greatly increase the accuracy of propagation loss density estimation.

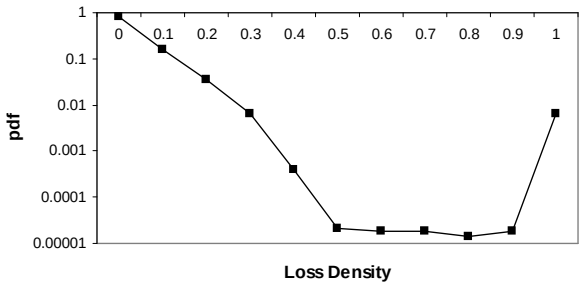

Fig. 7. Loss density distribution.

IV. Data Analysis

In this section, we examine the collected packet traces and show the analysis results of one typical trace group. First we choose a typical trace at one receiver to study the local packet loss distribution. Then we calculate the temporal loss pattern parameters at each receiver, such as average loss burst length and loss-free burst length. The results clearly show the effect of distance on overall packet loss rate. The spatial correlation is then studied. We calculate the correlation coefficient for each pair of receivers. The results as well as the distribution of loss density are presented.

A. Temporal Loss Characteristics

We choose a typical trace and plot it in Figure 8. In the figure, although the average loss rate is very low, the receiver experiences severe losses at certain time points. The loss burst can occasionally be as long as several hundred packets (not shown). However, loss bursts are only several packets long on average, and most of them are single packet losses. On the other hand, the loss free burst, or the distance between two consecutive loss bursts, is usually several hundred packets long. A few loss free bursts are even longer than 10,000 packets (also not shown). Therefore, applications running over wireless networks might experience no packet loss at all for a long period of time, but then lose many packets in a burst.

We plot the average loss rate, average loss burst length and average loss-free burst length at the 10 different receivers in Figure 9. The traces at different receivers exhibit different packet loss patterns. Basically, the distance from the AP determines packet loss rate at a receiver. The receivers that are more than 20 meters away from the AP exhibit poor performance, while there is little performance difference among receivers that are located within 10 meters from the AP. From Figure 9 we see that the average loss burst length is not correlated to the distance from the AP to the receiver. On the other

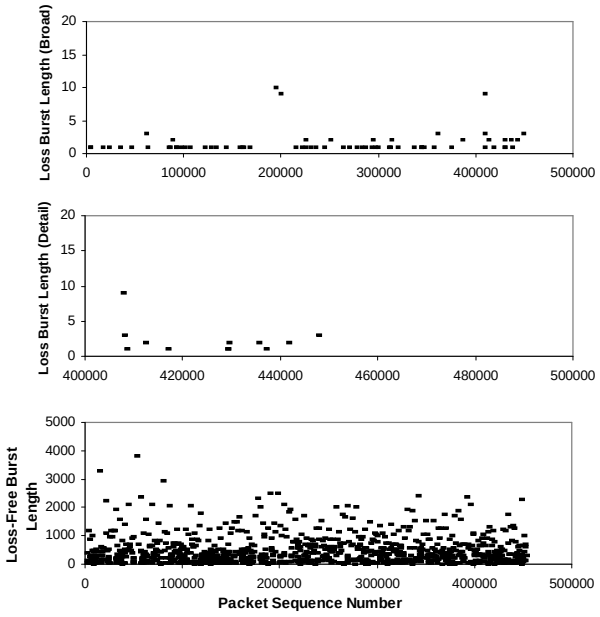

Fig. 8. Typical packet trace in wireless LAN.

hand, the average loss-free burst length is larger when the receiver is closer to the AP. In other words, loss-free burst length is inversely proportional to loss rate. Therefore, a high loss rate is mainly due to the high frequency of loss bursts instead of large loss bursts.

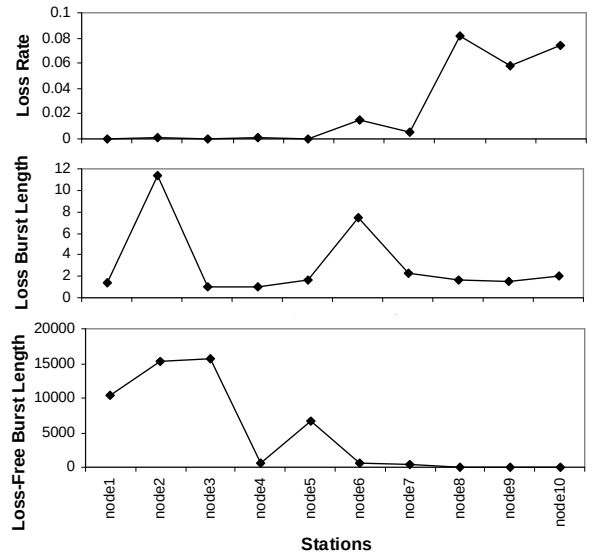

Fig. 9. Packet trace statistics.

Distance is not the only factor that affects the loss pattern at a station. The environment around a station determines the level of location-dependent random noise interference, thus it also affects the loss pattern. Sometimes the loss caused by interference may be more significant than the distance-dependent loss. We attribute the anomalous behavior in Figure 9 (such as the loss rates of nodes 6, 7, 8, the loss burst length at nodes 2 and 6, and the loss-free burst length at nodes 4 and 5) to such factors.

B. Spatial Loss Correlation

As described in [24], wireless loss is caused by distance-dependent signal degradation and location-dependent random interference. In a multicast group, loss patterns vary at different wireless stations because losses are distance- and location-dependent. On the other hand, in our experiments, as well as in all other one-to-many multicast applications, the packet source is the same for all receivers. When packets are transmitted from a wired station to wireless network through the AP, variable conditions at the AP such as transmission power level changes will affect the packet loss probability at receivers. Consequently, those losses are likely correlated.

We developed a program to compare packet-by-packet status (received or lost) in two traces. Let N be the total number of packets, X be the number of losses in the first trace, Y be the number of losses in the second trace, Z be the number of packets lost in both traces, we calculate the correlation coefficient of each trace pair using the following formula:

$$Coeff = \frac{\frac{Z}{N} - \frac{X}{N} \times \frac{Y}{N}}{\sqrt{\frac{X}{N} - \frac{X}{N} \times \frac{X}{N}} \times \sqrt{\frac{Y}{N} - \frac{Y}{N} \times \frac{Y}{N}}} = \frac{Z \times N - X \times Y}{\sqrt{X \times (N-X)} \times \sqrt{Y \times (N-Y)}}.$$

In Figure 10 we plot the correlation coefficient for each pair of traces. The figure shows that receivers located at similar distance from the AP exhibit relatively high loss correlation. This is due to the fact that they share a large number of distance-dependent losses. However, the absolute value of the correlation coefficient is low, which implies that, besides the variation of distance-dependent losses, there are quite a few losses that are location-dependent random losses. On the other hand, the receivers located at different distances from the AP exhibit very low loss correlation. In fact, the correlation coefficient is near zero, which means the losses are almost independent among those receivers.

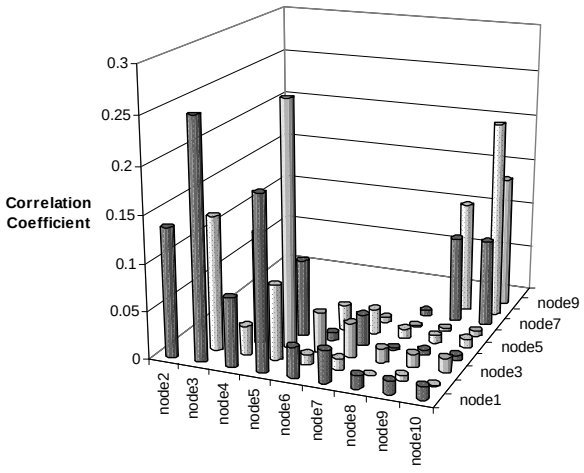

Fig. 10. Packet loss correlation.

To reflect the overall loss correlation in the group instead of in each pair, we calculate the loss density of the traces. Obviously, packet loss correlation has significant effect on this function. To show the effect, we compare the loss density of

the real traces with that of the traces generated by the independent model. We first consider an ideal case. Assume temporal packet loss distribution is Bernoulli, and spatial distribution is IID (Identical Independent Distribution). Let the loss rate at each receiver be p, and let the number of receivers be N. Then the probability of n out of N receivers simultaneously suffering a loss is given by the probability mass function of Binomial distribution:

$$P(n) = B(n, N; p) = C(N, n) \times p^n \times (1-p)^{N-n}.$$

Since the probability of at least one receiver suffering a loss is

$$\pi = 1 - (1-p)^N,$$

the ratio of n-receiver losses to all losses is equivalent to the conditional probability of an n-receiver loss occurrence given the fact that a packet is lost:

$$D(n) = \frac{P(n)}{\pi}.$$

This is also the probability mass function of loss density $\frac{n}{N}$. We set p to the mean of the average loss rates across all receivers. In Figure 11 we plot the distribution of loss density from the real traces and compare it with that of the analytical results. (The value corresponding to loss density 1 is absent in the real traces since the common losses have been excluded.)

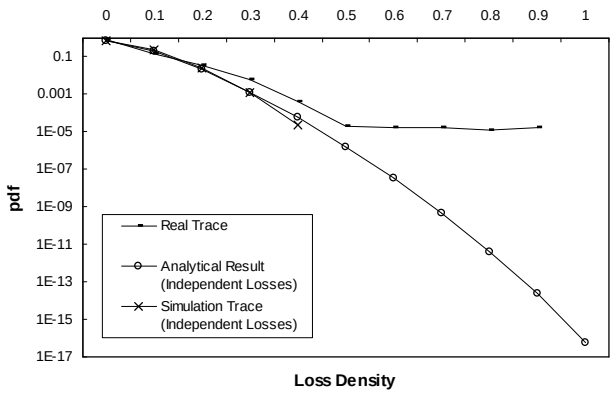

Fig. 11. Loss density comparison.

The naive assumption of temporally even distribution in the above analysis does not hold since we know wireless losses are bursty. To make a more accurate comparison, we generate individual traces from the Gilbert-Elliot model for each receiver, with the good-to-bad transition probability set to the reciprocal of the average loss free burst length, and the bad-to-good transition probability set to the reciprocal of the average loss burst length. We calculate the distribution of loss density from the generated trace and also plot it in Figure 11 (simulation trace with independent losses). The distribution shown is the average of five runs in which the random seeds of trace generation were varied. No values correspond to loss density above 0.4 because no such losses occur in the generated traces.

It is clear that the independent model significantly underestimates the probability of high loss density losses. Although the absolute value of the probability for those losses is very low, a single such occurrence may significantly affect the performance of multicast applications. As we saw in Figure 2, the increased service time can lead to an avalanche effect, especially when the multicast group is large. Figure 11 also implies that the loss density distributions are similar among the two independent models, regardless of the actual loss models used at the individual receivers. Therefore, the correlated model that we will describe in next section is orthogonal to any approaches aiming to improve modeling accuracy at individual stations.

V. Modeling Spatial Loss Correlation

In this section we introduce our approach of integrating spatial correlation constraint into the independent temporal packet loss models. We use loss density constraint for illustration. First we describe the general strategy. Then we present the procedure of loss density modeling. Next, three algorithms are proposed for correlating independent packet losses. The performance of these algorithms is compared and the statistics of the resulting correlated packet losses are presented.

A. The General Strategy

To comply with the loss density constraint, packet losses at receivers in a multicast group should not be independently modeled. The basic idea of our multi-stage approach is to independently generate packet traces for each receiver at first, using conventional models (for example, using Gilbert-Elliot model). The parameters of these models are set according to the location of the receivers, as well as to the interference scenario of the multicast transmission. The generated traces are then modified so that the overall group loss satisfies the loss density constraint.

Specifically, we first determine the loss density for each packet. Then the receivers that should experience packet loss are explicitly selected. Our goal is to minimize the chance of conflicts between the loss density requirement and the temporal models at individual receivers. The strategy is to rearrange the temporal order of either loss density random process or local packet loss bursts to avoid such conflicts. In the resulting traces, the loss density constraint is fully satisfied and the burst length distribution of each individual model is not significantly changed. On the other hand, the temporal order of bursts specified by the models may be compromised. For example, an application that is exposed to heavy losses at the beginning of its lifetime in independent model might suffer heavy losses at the end instead of the beginning in the correlated model due to rearrangement of loss bursts. However, these kinds of temporal order changes will unlikely have more significant impacts on the multicast applications than the loss density does. Moreover, most wireless loss models do not specify the modeling accuracy of loss temporal orders. Therefore, it is worthwhile to make this trade off.

Our approach supports any modeling strategy for both loss density and individual local packet losses. In fact, it operates on generated traces and does not directly involve mathematical modeling. As an example, we next describe how to generate a loss density distribution that matches the real loss density.

B. The Loss Density Distribution

The first step of our approach is to determine the loss density. We plot the CDF of the loss density of our traces in Figure 12. To find the distribution in the high loss-density section, we plot the function (1-CDF) on a logarithmic graph in Figure 13. Since our primary goal here is to show how the approach works, we use a straight line to approximate the (1-CDF) function on the logarithmic graph to avoid unnecessary complexity. Therefore, the loss density is modeled as an exponential distribution and the value of the parameter λ is obtained by applying regression. For this particular trace group, the value of λ is 15.6404. The resulting CDF is also plotted in Figure 12 and it matches relatively well to the real data. Now we can obtain the loss density of each packet from the formula

$$D = -\frac{ln(u)}{\lambda},$$

where u is a uniformly distributed random variable with values falling in the range of $e^{-15.6404}$ to 1.

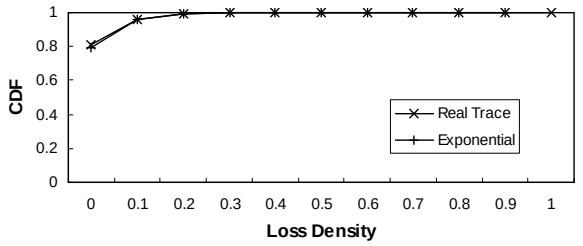

Fig. 12. Loss density CDF.

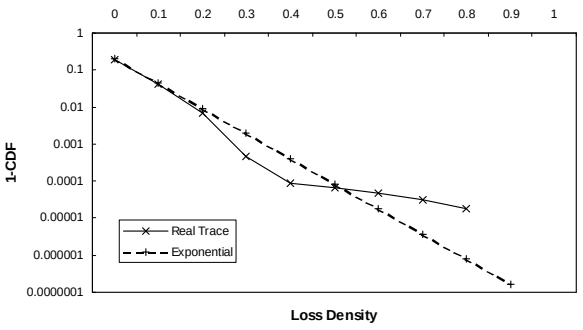

Fig. 13. log(1-CDF) of loss density.

C. Correlating Packet Losses

Next we select the receivers that are subject to loss of the current packet. Due to the constraint of loss density, it is possible that we have to set some receivers to loss-free state even if their local models demand them to be in loss state, or vice versa. This preemption of loss/loss-free state changes the length of current burst and affects the loss/loss-free burst length distribution as well as the local packet loss rate. Our goal is to minimize the impact of such changes.

To illustrate the behavior of our algorithms, we use an $n \times m$ matrix M to denote the global packet delivery status, where n is the number of total packets transmitted, and m is the number of receivers in the group. The value of the matrix element M_{ij} can be either 0 or 1, where 0 denotes the ith packet is successfully delivered to receiver j, while 1 means the packet is lost at receiver j. Let another $n \times m$ matrix M' be the global packet delivery status after the processing. Let $\vec{d}$ be the loss density vector and $\vec{w}$ be the $width$ vector. In other words, $\vec{d_i}$ denotes the loss density at the ith packet, and $\vec{w}_i$ and $\vec{w'}_i$ denotes the real or expected $width$, or equivalently, the number of receivers losing the ith packet. Formally, $\vec{w}_i = \sum_{j=1}^{m} M_{ij}$, and $\vec{w'}_i = \vec{d_i} \times m$. Let $\vec{a}_j$ denote the number of packet losses added to receiver j due to preemption, and let $\vec{c}_j = \sum_{i=1}^{n} M_{ij}$ be the total number of losses at receiver j. Moreover, let $\vec{l}_j = \frac{\vec{c}_j}{n}$ be the average packet loss rate at receiver j.

We propose three algorithms for correlating packet losses. The first is called Sequential Processing (SP) algorithm. This is a simple and straightforward algorithm. It processes packet losses sequentially using only current loss status. The second is called Loss Density Rearrangement (LDR) algorithm. This is an enhanced version of SP algorithm. It rearranges loss density values in order to minimize the number of affected packets. The third is called Loss Burst Rearrangement (LBR) algorithm. It rearranges loss and/or loss-free bursts in order to minimize the changes on burst length distributions. All these algorithms can generate packet loss traces that satisfy the loss density constraint. However, each algorithm has different advantages and disadvantages. Subsequently we describe these algorithms in detail.

1. Sequential Processing (SP) Algorithm

The SP algorithm works as follows: at each step i, the algorithm calculates the difference between the real and expected width $\vec{w}_i$ and $\vec{w'}_i$. If the real width is greater than the expected width, then we need to select $\vec{w}_i - \vec{w'}_i$ receivers in loss state and change their states to loss-free. Similarly, if the real width is less than the expected width, then we need to select $\vec{w'}_i - \vec{w}_i$ receivers in loss-free state and change to loss state. We always select the eligible receivers with higher $\vec{a}_j$ values for loss to loss-free transitions and lower $\vec{a}_j$ values for loss-free to loss transitions in order to minimize the changes to local packet loss rate. The algorithm is shown in Figure 14.

Algorithm SP
begin

Initialize M, $\vec{w}$, $\vec{w'}$, $\vec{a}$
$i = 0$
while $(i < n)$ {
  /* loop until all packets have been processed */
  $S \leftarrow \emptyset$
  if $(\vec{w}_i > \vec{w'}_i)$ {
    /* more receivers lost packet i than expected */
    /* loss to loss-free transition */
    for $(k = 0; k < \vec{w}_i - \vec{w'}_i; k++)$ {
      /* find $(\vec{w}_i - \vec{w'}_i)$ receivers that can change to loss-free state from loss state */
      if $(\exists j \in [1..m] \wedge \neg(j \in S) \wedge M_{i,j} == 1 \wedge$
        /* this is a loss-state receiver having not been considered */
        $(\forall j' \in [1..m], \neg(j' \in S) \Longrightarrow \vec{a}_{j'} \leq \vec{a_j})$
        /* this receiver has the least number of added losses */
) {

```
            S ← S ∪ j
              /* select receiver j for state transition */
            M_{i,j} = 0
              /* change from loss state to loss-free state */
            ā_j − −
              /* update the count of added losses for this receiver – removing a loss is considered as adding a minus number of loss */
          } /* end of if */
        } /* end of for */
      } /* end of if */
      else if (w⃗_i < w⃗'_i) {
        /* less receivers lost packet i than expected */
        /* loss-free to loss transition */
        for (k = 0; k < w⃗'_i − w⃗_i; k + +) {
          /* find (w⃗'_i − w⃗_i) receivers that can change to loss state from loss-free state */
          if (∃j ∈ [1..m] ∧ ¬(j ∈ S) ∧ M_{i,j} == 0∧
            /* this is a loss-free-state receiver having not been considered */
            (∀j' ∈ [1..m], ¬(j' ∈ S) ⟹ ā_{j'} ≥ ā_j)
            /* this receiver has the largest number of added losses */ ) {
            S ← S ∪ j
              /* select receiver j for state transition */
            M_{i,j} = 1
              /* change from loss-free state to loss state */
            ā_j + +
              /* update the count of added losses for this receiver */
          } /* end of if */
        } /* end of for */
      } /* end of if */
      i + + /* processing the next packet */
    } /* end of while */
end
```

Fig. 14. The Sequential Processing algorithm.

The major advantage of the SP algorithm is its simplicity. It does not need to maintain the entire global packet delivery status matrix M, and is fast on processing packet losses. Therefore, the SP algorithm is the most suitable for online processing. Its computational complexity is $O(nm)$. However, it may significantly change the local packet loss statistics in both packet loss temporal order and burst length distribution. Let c be the total number of rows where the expected width $\vec{w'}_i$ is non-zero. Then in the worst case, there are $c_j + c$ packets of which the loss state is toggled.

2. Loss Density Rearrangement (LDR) Algorithm

The LDR algorithm works as follows: it first sorts the real and expected width vectors $\vec{w}$ and $\vec{w'}$ in descending order, then processes packet losses in this order. At each step i, the algorithm selects the packet (row) corresponding to $\vec{w}_i$ and adjusts the loss status of this packet at each receiver using the same strategy as in the SP algorithm. It terminates when both $\vec{w}_i$ and $\vec{w'}_i$ becomes zero. The algorithm is shown in Figure 15

```
Algorithm LDR
begin

  Initialize M, w⃗, w⃗', ā
  Sort w⃗ and w⃗' in descending order, let Index(i) be the original row index of the ith row after sorting
  i = 0
  while (w⃗_i > 0 ∧ w⃗'_i > 0) {
    /* loop until all packets have been processed */
    /* if both w⃗_i and w⃗'_i are zero, then no further processing is needed since the subsequent w⃗_i and w⃗'_i are all zero */
    S ← ∅        if (w⃗_i > w⃗'_i) {
      /* more receivers lost packet i than expected */
      /* loss to loss-free transition */
      for (k = 0; k < w⃗_i − w⃗'_i; k + +) {
        /* find (w⃗_i − w⃗'_i) receivers that can change to loss-free state from loss state */
        if (∃j ∈ [1..m] ∧ ¬(j ∈ S) ∧ M_{Index(i),j} == 1∧
          /* this is a loss-state receiver having not been considered */
          (∀j' ∈ [1..m], ¬(j' ∈ S) ⟹ ā_{j'} ≤ ā_j)
          /* this receiver has the least number of added losses */ ) {
          S ← S ∪ j
            /* select receiver j for state transition */
          M_{Index(i),j} = 0
            /* change from loss state to loss-free state. */ Index(i) returns the row index corresponding to the current value of w⃗_i and w⃗'_i */
          ā_j − −
            /* update the count of added losses for this receiver – removing a loss is considered as adding a minus number of loss */
        } /* end of if */
      } /* end of for */
    } /* end of if */
    else if (w⃗_i < w⃗'_i) {
      /* less receivers lost packet i than expected */
      /* loss-free to loss transition */
      for (k = 0; k < w⃗'_i − w⃗_i; k + +) {
        /* find (w⃗'_i − w⃗_i) receivers that can change to loss state from loss-free state */
        if (∃j ∈ [1..m] ∧ ¬(j ∈ S) ∧ M_{Index(i),j} == 0∧
          /* this is a loss-free-state receiver having not been considered */
          (∀j' ∈ [1..m], ¬(j' ∈ S) ⟹ ā_{j'} ≥ ā_j)
          /* this receiver has the largest number of added losses */ ) {
          S ← S ∪ j
            /* select receiver j for state transition */
          M_{Index(i),j} = 1
            /* change from loss-free state to loss state. */ Index(i) returns the row index corresponding to the current value of w⃗_i and w⃗'_i */
          ā_j + +
            /* update the count of added losses for this receiver */
        } /* end of if */
      } /* end of for */
    } /* end of if */
    i + + /* processing the next packet */
  } /* end of while */
end
```

Fig. 15. The Loss Density Rearrangement algorithm.

Compared with the SP algorithm, the major advantage of the LDR algorithm is that the resulting delivery status M' is more similar to the original status M than that of the SP algorithm on packet loss temporal order. In other words, the number of packets whose loss state is toggled is minimized. This is due to the fact that we paired off the non-zero real and expected width in descending order so that in each row the number of

state changes is minimized and the number of rows that are affected is also minimized. On the other hand, the LDR algorithm is not sequential. It needs to know the width of all rows beforehand to sort the width vectors so that it is not suitable for online modeling. Moreover, it needs to maintain the entire global packet delivery status matrix M. The computational complexity is $O(n \log n + nm)$.

3. Loss Burst Rearrangement (LBR) Algorithm

The LBR algorithm rearranges the temporal order of loss/loss-free bursts in order to satisfy the loss density constraint. The algorithm works as follows: it maintains two preempted-burst lists, one for loss burst and another for loss-free burst, and two preempting-burst lists for each receiver. At first, it scans the packet loss states and generates a loss burst and a loss-free burst list at each receiver. Then it processes packets row by row like in the SP and LDR algorithms. At each step i, it determines how many receivers need to change their packet loss state and selects a set of receivers. The selection strategy will be described later. For each selected receiver, the effect of state change is expressed as the change of bursts. If the original state is loss, then the receiver is in a loss burst. Assume the loss burst starts at packet b, and the length of this burst is e, By changing the state to loss-free at i, we effectively replaced the loss burst of length e by a new loss burst of length $i - b$. We call the original burst as *preempted* burst and the new burst as *preempting* burst. To maintain the burst length distribution, we need to keep the preempted burst for later usage and match an unused original burst to the preempting burst. On the other hand, from the position i we started a new loss-free burst. We also need to select an unused original loss-free burst as this new burst in order to minimize the impact on burst length distribution.

We perform the burst management as follows: for the preempted loss burst B, we scan the preempting-loss-burst list at the receiver. If we find a burst B' in the preempting-loss-burst list that has the same length as the preempted burst B, which means a non-original loss burst B' with the same length has been used somewhere and we can image it is the preempted burst B being used there, then we do not need to keep the preempted burst B (since it has been used if we take B' as B), nor we need to keep B' anymore since it now becomes an "original" burst. So we simply mark B as "used" and remove B' from the preempting-loss-burst list. We call this process as "neutralization". If no such non-original burst is found, we put B into the preempted-loss-burst list.

For the preempting loss burst, we scan the preempted-loss-burst list for the chance of neutralization. If a matching burst is found, then it is removed from the preempted-loss-burst list and marked as "used", and the preempting loss burst is discarded. Otherwise, we scan the original loss burst list for the chance of neutralization, if the content of the entire list is available. If no such unused original loss burst is found, then we put the preempting loss burst into the preempting-loss-burst list.

For the new loss-free burst starting at the current position, we take the first burst in the preempted-loss-free-burst list. We remove this burst from the list, and use it as the new loss-free burst. We do not mark it as "used" at this time since it may be preempted again at the next step. If the preempted-loss-free-burst list is empty, then we take the first unused burst in the original loss-free burst list, and use it as the new loss-free burst. The whole processing is similar if we need to change the state from loss-free to loss at step i.

We can see from the above description, that the preempted-burst lists contain bursts that should have been used but have not been really used yet. On the other hand, the preempting-burst lists contain bursts that should not be used but have anyway been used somewhere. In order to maintain the burst length distribution, we should keep the size of the lists as small as possible. Therefore, we always select the receivers at each step that can reduce the size of their lists through the processing at this step. If no such receivers exist, then we select the receivers of which the sum of the size of the four lists is minimum in order to avoid buffer overflow. The LBR algorithm is shown in Figure 16.

Algorithm LBR
begin

Initialize M, $\vec{w}$, $\vec{w'}$, $\vec{a}$
Generate a loss burst list and a loss-free burst list for each receiver
$i = 0$
while ($i < n$**)** {
/* loop until all packets have been processed */
$S \leftarrow \emptyset$ **if (**$\vec{w}_i > \vec{w'}_i$**)** {
/* more receivers lost packet i **than expected */**
/* loss to loss-free transition */
for (each receiver j**)** {
/* check receivers one by one to see if it is eligible for state transition */
if ($\neg(j \in S) \wedge \mathrm{InLossBurst(i, j)} \wedge$ **(**
/* this is a loss-state receiver having not been considered */
PringB(j) $\in$ PredLBList(j) $\vee$
/* the current preempting burst is found in the preempted loss-burst list */
PredB(j) $\in$ PringLBList(j) $\vee$
/* the current preempted burst is found in the preempting loss-burst list */
$(\forall j' \in [1..m], \neg(j' \in S) \wedge j! = j' \Longrightarrow ListSize(j) \leq ListSize(j'))$
/* this receiver has the least number of in-list (suspended) bursts */
) {
$S \leftarrow S \cup j$
/* select receiver j for state transition */
} /* end of **if** */
if ($|S| == \vec{w}_i - \vec{w'}_i$**) break**
/* found enough receivers */
} **/* end of for */**
for (each receiver $j \in S$**)** {
/* process each selected receiver */
if (PringB(j) $\in$ PredLBList(j)) {
/* the current preempting burst is found in the preempted loss-burst list */
PredLBList(j)$\leftarrow$ PredLBList(j)- PringB(j)
/* remove the current preempting burst from the preempted loss-burst list */
} **else** {
/* the current preempting burst is a new burst not in the preempted loss-burst list */
PringLBList(j)$\leftarrow$ PringLBList(j)$\cup$ PringB(j)
/* add the current preempting burst to the preempting loss-burst list */

```
        } /* end of if */
        if (PredB(j) ∈ PringLBList(j)) {
            /* the current preempted burst is found in the preempting
loss-burst list */
            PringLBList(j)← PringLBList(j)- PredB(j)
            /* remove the current preempted burst from the pre-
empting loss-burst list */
        } else {
            /* the current preempted burst is not in the preempt-
ing loss-burst list */
            PredLBList(j)← PredLBList(j)∪ PredB(j)
            /* add the current preempted burst to the preempted
loss-burst list */
        } /* end of if */
        if (PredLFBList(j) == ∅) {
            /* the preempted loss-free-burst list is empty */
            CurLFB(j) ← LFB(j).next()
            /* select the next loss-free busrt from the original burst
list */
        } else {
            /* the preempted loss-free-burst list is not empty */
            CurLFB(j) ← PredLFBlist(j).pop()
            /* remove the first burst from the preempted loss-free-
burst list, and use it as the next loss-free burst */
        } /* end of if */
        Update(w⃗)
            /* update the row width vector since the selected burst
may have changed the status of subsequent rows */
      } /* end of for */
    } /* end of if */


    else if (w⃗_i < w⃗'_i) {
      /* less receivers lost packet i than expected */
      /* loss-free to loss transition */
      for (each receiver j) {
        /* check receivers one by one to see if it is eligible for
state transition */
        if (¬(j ∈ S) ∧ InLossFreeBurst(i, j)∧ (
            /* this is a loss-free-state receiver having not been
considered */
            PringB(j) ∈ PredLFBList(j) ∨
            /* the current preempting burst is found in the preempted
loss-free-burst list */
            PredB(j) ∈ PringLFBList(j) ∨
            /* the current preempted burst is found in the preempting
loss-free-burst list */
            (∀j' ∈ [1..m], ¬(j' ∈ S) ∧ j! = j' ⟹ ListSize(j) ≤
ListSize(j'))
            /* this receiver has the least number of in-list (sus-
pended) bursts */
            ) {
            S ← S ∪ j
              /* select receiver j for state transition */
        } /* end of if */
        if (|S| == w⃗'_i − w⃗_i) break
            /* found enough receivers */
      } /* end of for */
      for (each receiver j ∈ S) {
        /* process each selected receiver */
        if (PringB(j) ∈ PredLFBList(j)) {
            /* the current preempting burst is found in the preempted
loss-free-burst list */
            PredLFBList(j)← PredLFBList(j)- PringB(j)
            /* remove the current preempting burst from the
preempted loss-free-burst list */
        } else {
            /* the current preempting burst is a new burst not in
the preempted loss-free-burst list */
            PringLFBList(j)← PringLFBList(j)∪ PringB(j)
            /* add the current preempting burst to the preempting
loss-free-burst list */
        } /* end of if */
        if (PredB(j) ∈ PringLFBList(j)) {
            /* the current preempted burst is found in the preempting
loss-free-burst list */
            PringLFBList(j)← PringLFBList(j)- PredB(j)
            /* remove the current preempted burst from the pre-
empting loss-free-burst list */
        } else {
            /* the current preempted burst is not in the preempt-
ing loss-free-burst list */
            PredLFBList(j)← PredLFBList(j)∪ PredB(j)
            /* add the current preempted burst to the preempted
loss-free-burst list */
        } /* end of if */
        if (PredLBList(j) == ∅) {
            /* the preempted loss-burst list is empty */
            CurLB(j) ← LB(j).next()
            /* select the next loss busrt from the original burst list */
        } else {
            /* the preempted loss-burst list is not empty */
            CurLB(j) ← PredLBlist(j).pop()
            /* remove the first burst from the preempted loss-burst
list, and use it as the next loss burst */
        } /* end of if */
        Update(w⃗)
            /* update the row width vector since the selected burst
may have changed the status of subsequent rows */
      } /* end of for */
    } /* end of if */

    i + + /* processing the next packet */
  } /* end of while */
end
```

Fig. 16. The Loss Burst Rearrangement algorithm.

We illustrate the LBR processing in Figure 17. In the example, the loss density constraint demands exactly two out of the three receivers lose the current packet. However, only *receiver1* is currently in loss state. In this case, our algorithm selects another receiver , say *receiver2*, and preempts the current loss-free burst with a newly generated loss burst (with length 4). The expected and actual length of the loss-free burst are recorded in the preempted and preempting burst list at *receiver2* (5 and 2 respectively). As a result of the preemption, the current packet will be lost at both *receiver1* and *receiver2*. Next time when *receiver2* needs a new loss-free burst, it will pick up the recorded burst (with length 5) and remove it from the preempted loss-free burst list. When the list becomes empty and *receiver2* needs a loss-free burst again, it will use its local model to generate a new burst. If the length of the generated burst happens to be 2, the recorded burst with length 2 in the preempting loss-free burst list will be removed and the generated burst be discarded. In this case *receiver2* will use the local model to generate a new burst again.

Compared with the SP and LDR algorithms, the major advantage of the LBR algorithm is that the burst length distribution in the resulting packet traces does not change much. This implies that a receiver will experience similar bursts as it would do using the original packet loss trace. Moreover, the packet loss rate at each receiver is almost the same as in the original packet loss scenario. However, this algorithm does not aim to

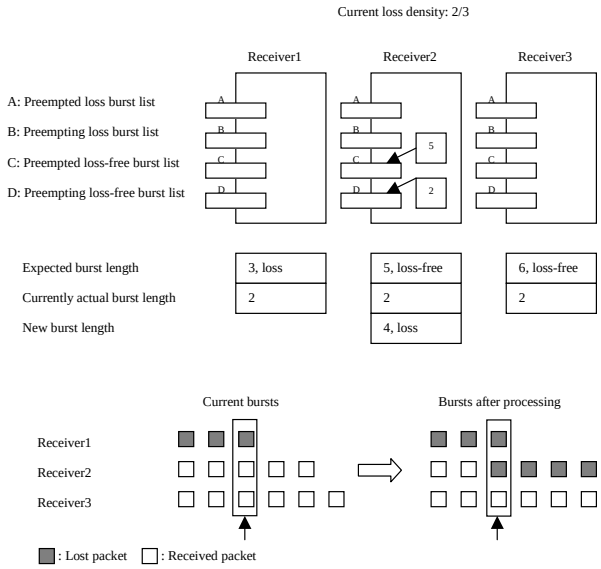

Fig. 17. Example of burst preemption in the correlated model.

maintain the temporal order of packet losses. Consequently, the resulting packet trace may look very different from the original trace although they may have the same burst length distribution and packet loss rate. Let s_p be the size of the preempted/preempting burst lists, and s_b be the number of original bursts, then the computational complexity of LBR algorithm is $O(nms_p + nms_b)$ if it scans the original burst list for preempting burst neutralization, and $O(nms_p)$ if it does not.

D. Evaluation

We evaluate the performance of our approach by comparing the loss density, average loss rate, average burst length, and burst length distribution before and after the processing. We also calculate the correlation coefficient between the original and processed traces. Our results show the processed traces can satisfy the loss density constraint while reasonably preserving the temporal characteristics in the original traces. Among the three algorithms, LBP is the best in preserving burst length distributions. On the other hand, LDP is good at achieving high temporal similarity with the original traces. Applications can choose different algorithms according to their specifications.

To conduct the evaluation, we first generate 10 independent loss traces using Gilbert-Elliot model. The state transition probabilities are obtained from the real traces. Then we run our algorithms to process the traces. After that, we compare the characteristics of the resulting traces with the original ones.

1. Loss Density

Our approach guarantees the resulting loss density matches the exponential distribution described in the previous section. This is verified by our resulting traces. The comparison of the loss density distribution is shown in Figure 18. In fact, all the three algorithms generated packet traces with exactly the same loss density distribution as expected. Therefore we only show the results from one algorithm. From the figure we can see that, even if we simply use an exponential curve to approximate the loss density distribution, the resulting model is much closer to the real traces, for high loss density values, than in the independent model.

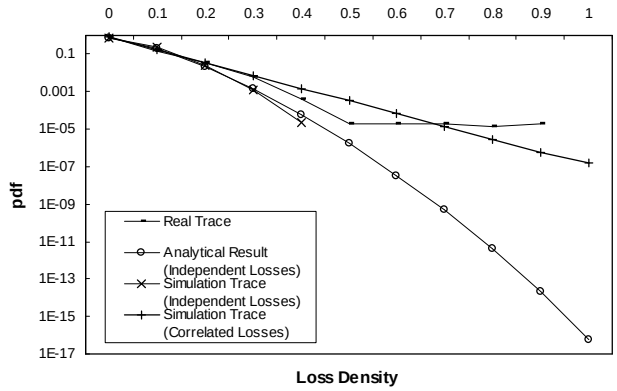

Fig. 18. Comparison of loss density distribution.

2. Average Loss Rate and Burst Length

To satisfy the loss density constraint, our algorithms have to frequently change the packet loss status at individual receivers. Therefore, the resulting average loss rate as well as the average loss/loss-free burst lengths may be different from that in the original traces. Figure 19 and Figure 20 quantify the effects on loss rate and burst length respectively. Figure 19 shows the impact on loss rate is insignificant, especially for the SP and LBR algorithms. The LBR algorithm processes packet traces by rearranging loss/loss-free bursts, therefore it does not really add or remove packet losses to the traces. Accumulatively the number of packet losses should be the same, as verified in the figure. Interestingly, SP has good performance on loss rate while LDP does not. It seems LDP tends to add packet losses to high-quality receivers while remove losses from those experiencing high losses. This is probably caused by the fact that LDP processes packets in the descending order of their loss density values. In this way, the high loss density rows are likely to be always paired with a higher expected loss density requirement. Therefore there should be more rows that need to add losses rather than remove losses. A high-quality receiver is more likely to be chosen for loss addition. On the other hand, it is also more likely to be chosen for loss removal if a receiver has already added a loss, since both the SP and LDR algorithms maintain an addition counter for each receiver. The random SP algorithm achieves good addition/removal balance for each receiver. However, if there are more additions than removals, as in the LDR algorithm, the high-quality receivers will have lower chance to counter-react to the added losses.

Figure 20 shows the reciprocal of average loss/loss-free burst length for each algorithm. Again the LBR algorithm is good at preserving the trace property of average burst length. The SP and LDR algorithms tend to produce shorter loss bursts, since a long burst is easily to be interrupted by these algorithms. Generally the LDR algorithm has better performance than the SP algorithm, especially at high-loss receivers.

3. Burst Length Distribution

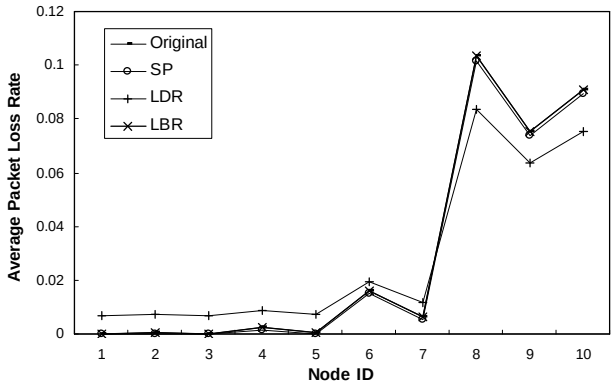

Fig. 19. Comparison of average packet loss rate.

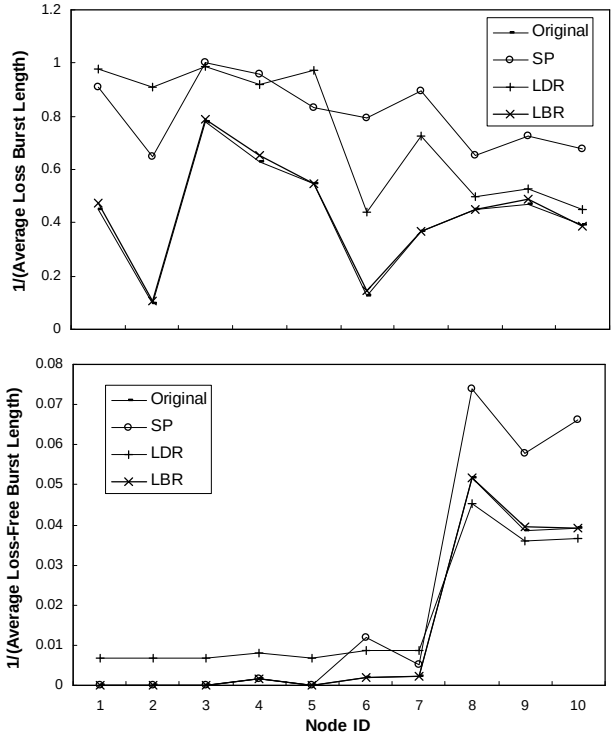

Fig. 20. Comparison of average loss/loss-free burst length.

Figure 21 shows the effects of our algorithms on the loss burst length distribution and the loss-free burst distribution. The presented data is for $node10$. Once again the LBR algorithm is proved to have good performance. The LDR algorithm also has comparable performance for loss burst length, but it seems the algorithm generated a few long loss-free bursts. This is probably due to the fact that LDR is less likely to split a long loss-free burst when compared to SP, where the random loss-free to loss transitions may generate a large number of small bursts, which can be observed in the figure. Generally speaking, although the burst length distribution might be slightly changed after the processing, the impact is not significant especially when the local model itself does not perform well on the burst length distribution modeling.

4. Temporal Similarity

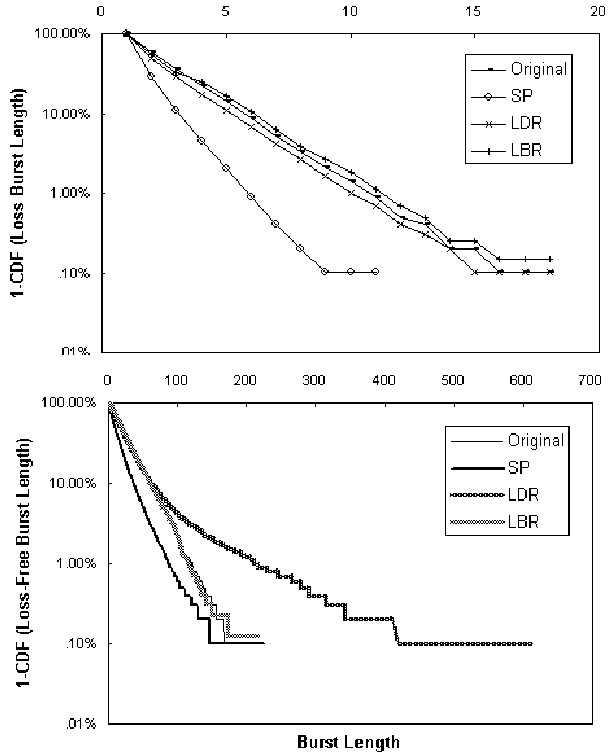

Fig. 21. Comparison of burst length distribution.

We calculate the correlation coefficient between the original and processed traces to show the temporal similarity between the pair. A higher correlation coefficient value denotes better-preserved temporal similarity. We can imagine this metric as the matched section proportion of two temporal loss curves. Figure 22 shows the correlation coefficient values corresponding to the three algorithms. It shows the LDR algorithm can achieve high temporal similarity, especially for high-loss receivers. This is expected since the algorithm only affects the loss status of a relatively small number of packets. On the contrary, the LBR algorithm does not have good performance on this metric. The burst rearrangement re-shuffles the loss status so that the temporal loss curve may be totally changed. The SP algorithm performs somewhere between LDR and LBR algorithms.

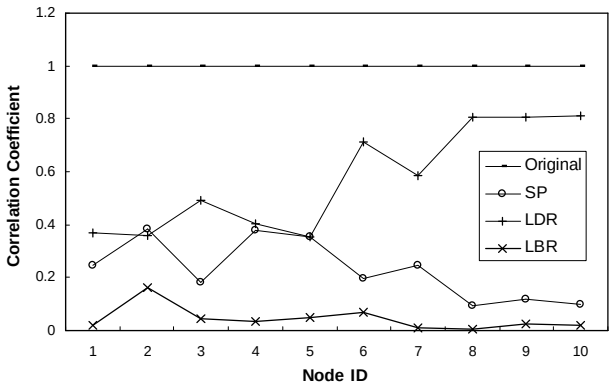

Fig. 22. Comparison of correlation coefficient between original and processed traces.

VI. Conclusion

In this paper we described our experience in measuring and modeling multicast packet losses in an IEEE 802.11 wireless LAN. We collected a large number of packet traces. When comparing the packet loss characteristics among traces in the same group, we observed that the packet losses at multiple receivers are often correlated. We introduced the concept of loss density, which we use to measure the degree of the correlation. Lastly, we proposed an approach to model multicast packet losses in wireless LANs that takes this spatial loss correlation into account. This approach is likely orthogonal to other approaches that aim to improve the modeling accuracy for individual stations. Simulation results show the proposed approach is more accurate than the conventional independent loss approach in modeling multicast packet losses. This approach can also be applied in other environments where both temporal and spatial packet loss characteristics should be considered.

While the improvement in modeling accuracy is shown for the collected traces, the significance and the validity of the approach in networks with different configurations needs to be evaluated. The modeling of the loss density distribution could be further enhanced (for example, split the loss density curve into two segments and use different exponential distribution to model them). Finally, the effect of the improved model on performance evaluation of real multicast applications requires investigation.

References

[1] D. Towsley, J. Kurose, and S. Pingali, "A comparison of sender-initiated and receiver-initiated reliable multicast protocols," *IEEE Journal on Selected Areas in Communications 15, 3*, pp. 398–406, April 1997.

[2] P. Ge and P. K. McKinley, "Experimental evaluation of error control for video multicast over wireless LANs," in *Proceedings of the Third International Workshop on Multimedia Network Systems*, (Phoenix, Arizona), April 2001.

[3] M. Yajnick, J. Kurose, and D. Towsley, "Packet loss correlation in the MBONE multicast network," in *Proceedings of IEEE Global Internet, November 1996*, November 1996.

[4] J. Andersen, T. Rappaport, and S. Yeshiva, "Propagation measurements and models for wireless communications channels," *IEEE Communications Magazine*, vol. 42-49, 1995.

[5] D. Eckhardt and P. Steenkiste, "Measurement and analysis of the error characteristics of an in-building wireless network," in *Proceedings of ACM SIGCOMM*, 1996.

[6] H. Wang and N. Moayeri, "Finite state markov channel - a useful model for radio communication channels," *IEEE Transactions on Vehicle Technology*, pp. 163–171, February 1995.

[7] A. Konrad, B. Y. Zhao, A. D. Joseph, and R. Ludwig, "A markov-based channel model algorithm for wireless networks," in *Proceedings of ACM MSWiM*, 2001.

[8] M. Zorzi and R. Rao, "On channel modeling for delay analysis of packet communications over wireless links," in *Proceedings of 36th Annual Allerton Conference*, September 1998.

[9] Y. Mohasseb and M. Fitz, "A 3-d spatio-temporal simulation model for wireless channels," *IEEE Journal on Selected Areas in Communications*, vol. 20, pp. 1193–203, August 2002.

[10] I. Viering, M. Reinhardt, and T. Frey, "Statistical modelling of spatially correlated MIMO channels," in *Proceedings of IEEE International Symposium on Intelligent Signal Processing*, 2001.

[11] D. Duchamp and N. Reynolds, "Measured performance of wireless LAN," in *Proceedings of 17th Conference on Local Computer Networks*, 1992.

[12] E. Gilbert, "Capacity of a burst-noise channel," *Bell System Technology Journal*, vol. 39, pp. 1253–1265, September 1960.

[13] E. Elliot, "Estimates of error rates for codes on burst-noise channels," *Bell System Technology Journal*, vol. 42, pp. 1977–1997, September 1963.

[14] R. Jain, *The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling*. John Wiley & Sons, April 1991.

[15] G. T. Nguyen, R. Katz, and B. Noble, "A trace-based approach for modeling wireless channel behavior," in *Proceedings of the Winter Simulation Conference*, pp. 597–604, 1996.

[16] M. Mosko and J. Garcia-Luna-Aceves, "An analysis of packet loss correlation in FEC-enhanced multicast trees," in *Proceedings of 8th IEEE International Conference on Network Protocols*, November 2000.

[17] D. Avidor and S. Mukherjee, "Hidden issues in the simulation of fixed wireless systems," in *Proceedings of the 2000 Applied Telecommunication Symposium*, 2000.

[18] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, 1984.

[19] IEEE, "IEEE Standards for Information Technology – Telecommunications and Information Exchange between Systems – Local and Metropolitan Area Network – Specific Requirements – Part 11: Wireless LAN Medium Access control (MAC) and Physical Layer (PHY) Specifications," *IEEE 802.11, 1999 Edition (ISO/IEC 8802-11: 1999)*, 1999, http://standards.ieee.org/getieee802/802.11.html.

[20] C. Tang, "Adaptive Reliable Multicast in Wireless Local Area Networks," Master's thesis, Michigan State University, 2002.

[21] Y. Chawathe, S. Fink, S. McCanne, and E. Brewer, "A proxy architecture for reliable multicast in heterogeneous environments," in *Proceedings of ACM Multimedia '98*, (Bristol, UK), September 1998.

[22] P. K. McKinley and A. P. Mani, "An experimental study of adaptive forward error correction for wireless collaborative computing," in *Proceedings of the IEEE 2001 Symposium on Applications and the Internet (SAINT-01)*, (San Diego-Mission Valley, California), January 2001.

[23] A. Kamerman and G. Aben, "Net Throughput with IEEE 802.11 Wireless LANs," in *Proceedings of IEEE Wireless Communications and Networking Confernce (WCNC)*, pp. 747–752, 2000.

[24] Y. Xu and T. Zhang, "An adaptive redundancy technique for wireless indoor multicasting," in *Proceedings of IEEE Symposium on Computers and Communications*, July 2000.