Dimensioning Links for IP Telephony

Bengt Ahlgren, Anders Andersson, Olof Hagsand and Ian Marsh
SICS
CNA Laboratory
Sweden
{bengta, olof, andersa, ianm}@sics.se

Abstract—

Packet loss is an important parameter for dimensioning network links or traffic classes carrying IP telephony traffic. We present a model based on the Markov modulated Poisson process (MMPP) which calculates packet loss probabilities for a set of superpositioned voice input sources and the specified link properties. We do not introduce another new model to the community, rather try and verify one of the existing models via extensive simulation and a real world implementation. A plethora of excellent research on queuing theory is *still* in the domain of ATM researchers and we attempt to highlight it's validity to the IP Telephony community.

Packet level simulations show very good correspondence with the predictions of the model. Our main contribution is the verification of the MMPP model with measurements in a laboratory environment. The loss rates predicted by the model are in general close to the measured loss rates and the loss rates obtained with simulation. The general conclusion is that the MMPP-based model is a tool well suited for dimensioning links carrying packetized voice in a system with limited buffer space.

Keywords—**Link Dimensioning, Markov Process, IP Telephony, MMPP/D/1/K**

I. Introduction

Voice applications, such as telephony, have been used on the best effort service provided by the Internet for quite some time. Currently many telephone operators have advanced plans to use IP technology as a bearer also for the regular telephone service. This, however, requires that the IP network can provide service guarantees.

Quality of Service (QoS) issues are being addressed by many forums, committees and researchers. Research on IP QoS has concentrated on the issues of classifying, scheduling and admission of packets into a network. Less has been done on how to dimension an IP network carrying real time traffic.

This paper focuses on dimensioning IP network links intended to carry packetized telephony or voice calls. It is feasible that existing carriers would like to allocate a portion of their bandwidth for this service and through mechanisms like differentiated services [11] provide superior service for this kind of data and subsequently levy higher charges.

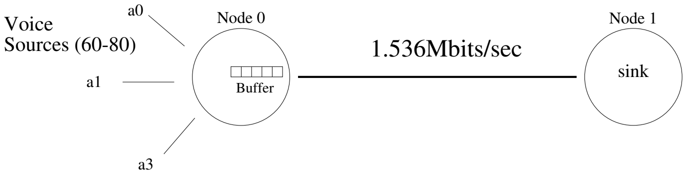

Fig. 1. Problem: dimensioning a link for voice sources over IP.

Our approach is to look at work done in *both* the ATM and traditional telephony communities as well as to use tools and simulators from the IP community to verify these ideas in an environment relevant for the Internet today. We have seen very little work which has taken this approach. The research community is divided into one of the two camps (but is changing as ATM and telephony people are more engaged in Internet research now).

Figure 1 illustrates the problem scenario we are addressing. A number of packet voice sources are multiplexed onto a link. The link has a limited amount of buffering which sometimes will result in the loss of packets with the obvious consequences on sound quality. With a link of a given bandwidth and a number of voice sources, what kind of quality could be expected if we ran 60 sources? What if we increased to 80—can we still expect adequate quality? How will we affect the system by changing the amount of buffering in the router?

We present a mathematical model based on a Markov modulated Poisson process (MMPP) which can predict the packet loss probability. We first verify the model using the NS packet level simulator. The main contribution of this paper is the verification of the MMPP model with measurements in a lab network. These experiments show a very good correspondence between the loss rate predicted by the model and the loss rate measured in the lab.

The rest of the paper is organized as follows. After summarizing relevant related work in the next section, we present the MMPP-based mathematical model and the reasoning leading to this model in Section III. Section IV describes the parameters we used in the experiments. Sections V and VI describe the NS simulations and the laboratory experiments, respectively. The experimental results are presented and discussed in Section VII and the paper is concluded with Section VIII.

II. Related work

Link dimensioning for voice has been a research topic for several decades in both academia and the telecommunications industry. Starting a little more than ten years back, the research focus has been on link dimensioning for ATM networks. Most of the results in the domain of ATM networks are also applicable in the domain of IP networks, since both are packet switching systems. The majority of the results from previous research is theoretical or results from simulations. Our research also has results from measurements of a real system.

Several approaches have been suggested in the literature to solve the problem of dimensioning links in packet switched networks. Anick, Mitra and Sondhi [2] study a multiplexer with infinite buffer with a *stochastic fluid flow* model but it is shown by Zheng [14] that this model only works for a multiplexer under heavy load. Tucker [15] studies a multiplexer with finite buffer using the fluid flow model, but it does not fit the model well for small buffers. Heffes and Lucantoni [7] uses a two-state *Markov modulated Poisson process* (MMPP) quite successfully to estimate the delay in a multiplexer with infinite buffer size. They suggest that the same approach for calculating the parameters of the MMPP can be used for a multiplexer with finite buffer size, but Nagarajan, Kurose and Towsley [10] show that this does not work in the case of finite buffer size. Instead, they develop a different method for finding the parameters of the MMPP. Baiocchi *et al.* [4] approximate the arrival process with a two-state MMPP and suggest a method called *asymptotic matching* for the calculation of the parameters of the MMPP. This approach is used by Andersson [1] together with a procedure to calculate the loss probabilities developed by Baiocchi, Melazzi and Roveri [3] to study a multiplexer loaded with a superposition of voice sources.

III. Mathematical model

In this section we develop a mathematical model for dimensioning a link carrying voice traffic. We start with the arrival process of a single IP telephony source and proceed with the superposition of independent identically distributed sources. The sources are then multiplexed on a bottleneck link through a queue of limited size. A more detailed description of this model can be found in previous work by one of the authors [1]. The model is based on a model developed by Baiocchi, Melazzi and Roveri [3].

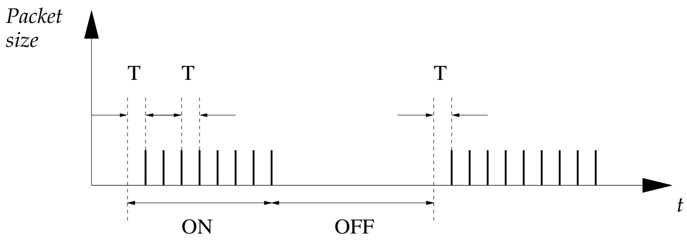

Fig. 2. Characteristics of a single source.

A. Single source properties

Most standard voice encodings have a fixed bit rate and a fixed packetization delay. They are thus producing a stream of fixed size packets. This packet stream is however only produced during talk-spurts—the voice coder sends no packets during silence periods.

The behavior of a single source is easily modeled by a simple on-off model (Figure 2). During talk-spurts (ON-periods), the model produces a stream of fixed size packets with fixed inter-arrival times T. Note that the first packet is produced one packet time after the start of an on-period. This is the result of the packetization—the voice coder has to collect voice samples before it can produce the first packet.

The number of packets in a talk-spurt, denoted with the stochastic variable N_b, is assumed to be geometrically distributed on the positive integers with mean n. This means that we can never have zero packets in a talk-spurt. This variant of the geometric distribution is sometimes called *first success* distribution (see for instance Gut [6, page 258]), and has the probability function:

$$P(N_b = k) = qp^{k-1}, k = 1, 2, 3, \dots \quad (1)$$

where q represents the probability that a packet is the last one in a talk-spurt. This means that $p = \frac{n-1}{n}$. This fact implies that the ON-periods have a expected value of $\alpha = nT$, where n is the expected value of the number of packets in a talk-spurt.

We assume that the OFF-periods are exponentially distributed with mean β, which is well documented and discussed by Sriram and Whitt [13]. A voice source may be viewed as a two state birth-death process with birth rate β and death rate α. The OFF state represents the idle periods and the ON state represents the talk-spurts. While in a talk-spurt, packets are generated with a rate of $\frac{1}{T}$ packets per second.

B. Approximating the single source

We have chosen to approximate the above model using exponentially distributed inter-arrival times with mean T

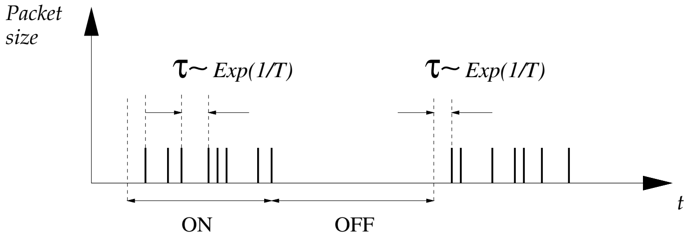

Fig. 3. A single source approximated with exponentially distributed inter-arrivals.

instead of fixed inter-arrival times. The purpose of the approximation is to simplify the modelling of many sources.

We let $\tau \in Exp(\frac{1}{T})$ denote the stochastic variable which describes the inter-arrivals during talk-spurts, and N_b be the geometrically distributed stochastic variable with the probability function stated in Equation 1 with mean n describing the number of packets in a talk-spurt. Moreover τ and N_b are assumed to be independent. It can be easily seen that the ON-periods (denoted U) are exponentially distributed and that the mean length of a talk-spurt is the same as in the deterministic inter-arrival case (nT). Figure 3 illustrates the behaviour of a single source with exponentially distributed inter-arrivals.

As in the previous section the OFF-periods are assumed to be exponentially distributed with mean β. Because of the exponentially distributed inter-arrival times during a talk-spurt, the emission of packets during an ON-period can be regarded as a Poisson process with intensity T. We can use the two state birth-death process to describe the packet generation with one state representing the idle periods and the other state representing the talk-spurts where packets are generated as a Poisson process with intensity T.

C. The superposition of independent voice sources

The superposition of voice sources can be viewed as a birth-death process where the states represent the number of sources that are currently in the ON-state. Here state i represents that i sources are active in a talk-spurt. We refer to the birth-death process as the *phase process* $J(t)$. The birth rate is given by the mean of the exponentially distributed idle periods, and we denote the mean as $\frac{1}{\beta}$. The death rate is determined by the mean of duration of the talk-spurts and is denoted $\frac{1}{\alpha}$. The probability p_{on} that a source is on is given by:

$$p_{\text{on}} = \frac{\alpha}{\alpha + \beta}.$$

D. Markov modulated Poisson process

The *Markov modulated Poisson process* (MMPP) is a widely used tool for analysis of tele-traffic models (see,

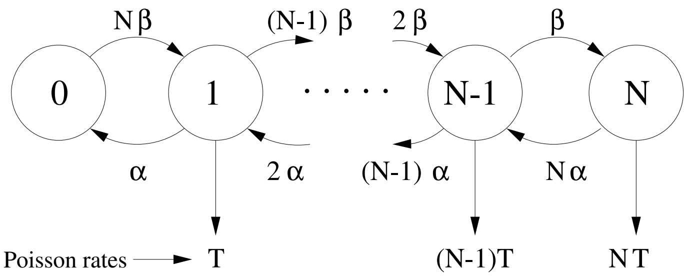

Fig. 4. Superposition of N voice sources with exponentially distributed inter-arrivals.

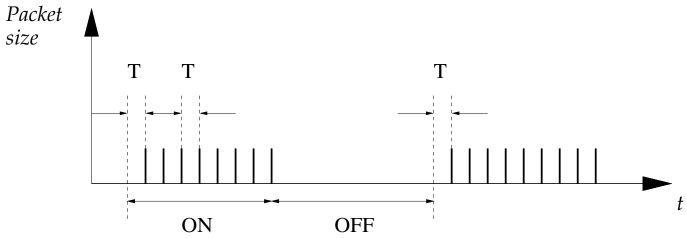

Fig. 5. k-interval squared coefficient of variation curves for superposition of N voice sources.

e.g., Heffes and Lucantoni [7]). It describes the superposition of sources of the type described in Section III-B. When the phase process is in state i, i sources are on. The model graph of the MMPP is shown in Figure 4.

The superposition of Poisson processes is also a Poisson process. We can therefore simply add the intensities of the sources that are currently in a talk-spurt and receive a new Poisson process for the superposition.

To validate the accuracy of approximating with a MMPP process, we calculated the index of dispersion of intervals (IDI) using a formula from Sriram and Whitt [13]. The IDI, also called the squared coefficient of variation, gives us some measure of how similar the traffic is in terms of burstiness. A value of 1 shows the traffic is as bursty as Poisson traffic, whereas a value as 18 is the burstiness of a single voice source. The high value accounts for the fact that the source is indeed bursty. The time period under which one observes this behaviour is very important.

Figure 5 shows c^2_{kN}, the IDI, versus k for k between 1 and 2000 and the number of sources, N, equal to 1, 10, 60 and 130. As a reference we have added the value of c^2_{kN} for a Poisson process. Data was obtained from simulations using a Matlab program. The solid line shows the c^2_{kN} for sources with deterministic inter-arrival times between packets during a talk-spurt, and the dashed lines show the c^2_{kN} for sources with exponentially distributed inter-arrival times, i.e., the MMPP approximation.

We see in the figure that the two descriptions of a single source behave in a similar way when they are superpositioned. The figure also shows that the superpositioned arrival process behaves as a Poisson process if we look at it for a short instant of time but it is much burstier if we study it over a longer period of time.

E. The multiplexer: MMPP/D/1/K queue

The arrival process described by the MMPP model is fed into a simple D/1/K queue. It is deterministic, has a single FIFO server and a buffer size (waiting room) which we vary. This kind of model is described in detail by Baiocchi *et al.* [3], [4]. We use their method and formulas for calculating the loss probability.

IV. Parameter values

We used the following parameters to run the MMPP model, simulations and lab experiments:

- 32 kb/s ADPCM voice encoding with 16 ms packet interarrival time, which results in 64 bytes of voice payload per packet
- A protocol header overhead consisting of 12 bytes for RTP, 8 bytes UDP and 20 bytes IP. We do not include any link layer headers. The resulting total packet size is 104 bytes, and the resulting bit rate is 52 kb/s.
- The number of successive packets in one talk-spurt is geometrically distributed on the positive integers with a mean of 22, which results in a mean talk-spurt length of 352 ms. The idle time between two successive bursts is exponentially distributed with a mean of 650 ms. The resulting average fraction of time a source is in a talk-spurt is 0.351.
- The bottleneck is a T1 link with a bandwidth of 1.536 Mb/s.

These values coincide with Sriram and Whitt [13] as well as previous work done by Zheng [14] whilst at SICS and Andersson [1], except that we in this paper include protocol header overhead for the RTP/UDP/IP protocol stack.

Figure 6 shows loss curves computed with the MMPP model for a sample set of buffer sizes. The next steps are to compare these loss probabilities from the model with results from NS simulations and measurements from a lab network.

A. Load

We use between 60 and 80 sources to load the link. To define a load that is independent of the link bandwidth the load factor, or λ, is used in the literature:

$$Load\ (\lambda) = \frac{N \times P_{\mathrm{on}} \times Rate_{\mathrm{peak}}}{C}$$

where N is number of sources, C is the link capacity, P_{on} is the probability that the source is on and $Rate_{\mathrm{peak}}$ speaks for itself. Table I shows loads for different numbers of sources.

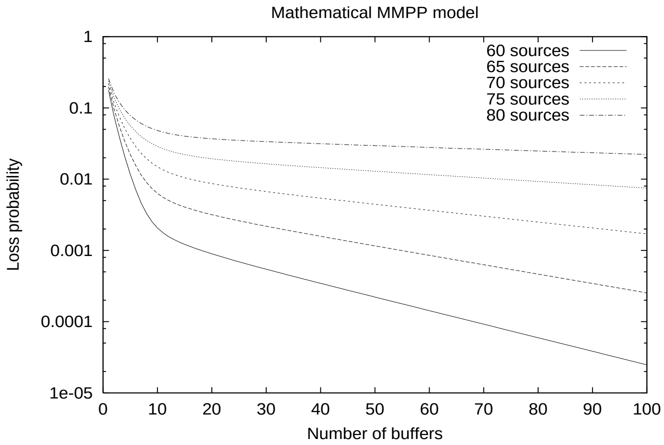

Fig. 6. Loss probabilities computed with the MMPP model.

Sources (N)	Load (λ)
29	34.5 %
60	71.4 %
80	95.3 %
84	98 %

TABLE I
Network load for a number of sources.

We decided to run between 60 and 80 sources as 84 sources is where the mean bandwidth of the sources equals to the bandwidth of the link. The peak allocation is as low as 29 sources (100 % utilisation when $P_{\mathrm{on}} = 1$) so taking advantage of the probability that a source is off yields much higher link utilisation.

B. Buffer size

We have chosen to simulate a multiplexer with an output link capacity of 1.536 Mb/s and buffer sizes ranging from 2 to 100 packets. With this choice of parameters we introduce a maximum queueing delay of 54 ms in the buffer. According to ITU recommendation G.114 [8] a delay of 0-150 ms acceptable for telephony, between 150 and 400 ms can also be acceptable, but over 400 ms is not. The total acceptable delay must be divided into a delay budget for each node in the path between the sender and receiver. If the path has 15 hops, and half of the delay budget can be allocated to queueing delay, then we get 13.3 ms per hop. This translates to approximately 24 buffers per hop. For higher bandwidth links, the queueing delay per buffered packet decreases inversely proportional to the bandwidth.

```
set cbr($i) [new Agent/CBR/UDP]

set exp($i) [new Traffic/Expoo]
$exp($i) set packet-size 104
$exp($i) set burst-time 0.352s
$exp($i) set idle-time 0.65s
$exp($i) set rate 52K

$cbr($i) attach-traffic $exp($i)
```

Fig. 7. Tcl code fragment defining a source NS-2.

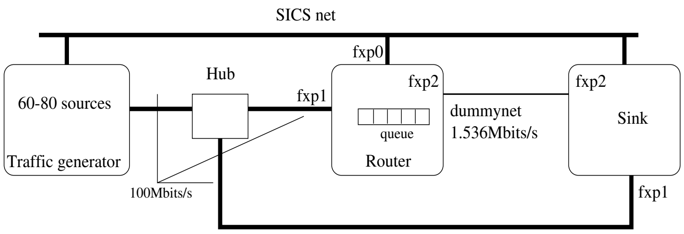

Fig. 8. Topology for Laboratory. The outgoing interface of the router is also connected to the sink.

V. NS SIMULATION

We used ns-2 [5], a packet level simulator to verify the MMPP model. Figure 1 shows the topology used in the simulations and Figure 7 the Tcl code that is used to start "agents". They are constant rate sources, denoted by "CBR/UDP". Traffic/Expoo generates traffic based on an exponential on/off distribution with the parameters specified in the next four lines. Each CBR source `$i$` uses a different random number seed, hence the sources will start independently of each other.

The simulation should run long enough for the system to reach steady state, ideally the system should be run for an infinite amount of time, however this is not practical due to time and resource constraints. A reasonable trade-off is to use a simulated time of 1000 seconds in both the simulation and the lab experiments. 1000 seconds with an interval of 16 ms generates 22000 packets per source and 1.32 million packets for 60 sources or 1.76 million for 80 sources.

VI. LAB NETWORK MEASUREMENTS

A. Topology

Figure 8 shows the experimental setup. A single machine acts as a traffic generator and emulates several IP Telephony 'calls' multiplexed together. The traffic is then sent on a shared 100 Mb/s Ethernet and received by two hosts: (1) a machine configured as a router; (2) a sink machine for measurement purposes. An outgoing link of the router is connected to the sink. In this configuration the traffic is emitted by the generator, passes through the router and is received by the sink. Since the sink can observe the packets before it enters the router, it can directly compare latency and loss of each individual packet. The outgoing link of the router is constrained to 1.536 Mb/s using Dummynet [12] which is explained in the next section. All the machines in the experiment were running FreeBSD 3.4.

B. Dummynet

Dummynet is a link emulator which allows arbitrary bandwidths and latencies to be specified. It is often used for emulating a slower link than what is physically available. Buffer sizes can be set for a given link and loss rates set to emulate the effect of lossy links. It is possible to create the illusion for TCP/UDP and IP that the link is like a WAN rather than a LAN. We are primarily interested in the lower bandwidth and configurable queue sizes. We modified the output functionality slightly to enable simpler calculation of the total number of packets received as well as the drop rate. Recording the total number of packets received gives us an additional check if the traffic generator or any system component lost/dropped packets during the experiment.

The total number of sent packets remained the same for a given source count and can be checked with the output of the traffic generator. It is trivial with a script to divide the loss by the total number of packets to obtain the loss rate.

C. Packet capture

To verify the loss rate we gathered the packets on the sink machine via a program that we developed[1] using the Berkeley Packet Filter [9] . Figure 8 shows that the output of the generator is attached directly to the sink machine as well as the outgoing link of the router. This enables us to capture all the packets and the ones not dropped by the router. A simple difference between the two should verify the loss rate reported by Dummynet. Our bpf program captures packets with a specific destination and port, and prints the time of arrival, RTP `src` and `seq` fields.

D. Traffic generator

The idea of the traffic generator is to create a sequence of packets that resemble many individual IP telephony calls multiplexed together. Furthermore, it should perform this job as accurately as possible with each packet emerging with a given deadline.

[1] Not tcpdump. We wrote out our own kernel filter to extract the packets we wanted as well as a user space program to output headers from 2 interfaces simultaneously.

```
#define INVERSE_M ((double)   4.6566128200e-10)
/* little number */

int calc_length(double burstlen) {
  double rand, logvalue;

  rand = INVERSE_M * random();
  logvalue = burstlen * -log(rand);

  return ((int)(logvalue + 0.5));
}
```

Fig. 9. C code to “randomize” a burst length

```
              source
      0 1 2 3 4 5 6 7 8 9
time
0     0 1 0 1 0 1 0 1 0 1
1     0 1 1 0 1 1 0 1 1 1
2     1 1 1 1 1 1 0 1 0 1
3     1 0 1 0 1 0 1 0 0 1
4     1 0 1 0 1 0 1 0 0 1
5     0 1 1 0 1 1 0 1 1 0
```

Fig. 10. Traffic generator trace file.

D.1 Trace file generation and playback

In order to be able to repeat experiments, we first precalculate the sending times of the packets and generate trace files. These files are then fed into the traffic generator which sends packets according to the trace. The trace files also allow us to test our setup to see if packets were being generated at the right times (such as inter-arrival times and sequence). The files are generated on a per source basis. The average length of a burst is calculated as shown in Equation 2.

$$burst\ length = \mathrm{rand}\left(\frac{P_{\mathrm{on}}}{interval}\right) \qquad (2)$$

The C-code for the rand function is shown in Figure 9.

Using the logarithm of the random variable generates burst lengths which are exponentially distributed.

The same calculation is applied for the idle (with P_{off}) period. The result is (reading vertically for each source) an exponentially distributed series of ON and OFF sequences with a mean ON of 0.351 seconds, OFF of 0.65 seconds which results in a burst length of 22 packets. An example of a trace file[2] with ten sources is shown in Figure 10.

The file shows for each time step (in this case 16 ms) which of the 10 sources are on or off. In the example, sources 1, 3, 5, 7 and 9 sends packets in the first time step. The traces of one source can be followed by reading a column downwards. Source 2, for example, sends no packet

[2]Actually it is converted into a binary format for more compact representation

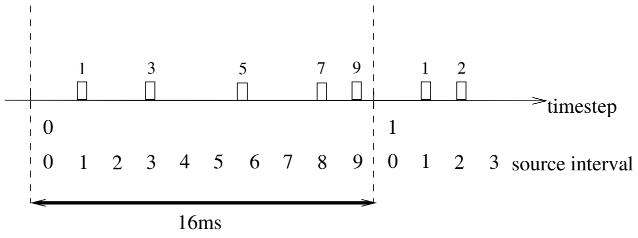

Fig. 11. Traffic generator sending times

in the first timestep, but then sends a packet in each of the succeeding steps.

If there are n sources, each timestep is further subdivided into n sub steps. Each sub step defines the sending interval for each source. For example, with ten sources and a time step of 16 ms starting at t, source 0 sends its packet within $[t, t + 1.6]$; source 1 sends within $[t + 1.6, t + 3.2]$, etc. If a source does not send its packets within its interval, it is said to miss its deadline. Packets that miss their deadline are recorded by the generator and printed when the run has completed as well as the largest value by which a packet was delayed.

So for the trace file above, the first steps of a packet sequence is shown in Figure 11. The sending of each packet is depicted as a horizontal interval, corresponding to the entering and leaving of the send system call, respectively. In the picture, the packets of source 5 and 7 missed their deadlines. The actual sending time on the link can be measured by an external mechanism, such as the packet capture program described previously.

D.2 Traffic generator verification

As a simple test for a trace file of 220000 packets we obtained values 36.9 % for the on time, 63.1 % for the off time by simply counting the ones and zeros in one column of the file. The mean number of packets in a burst equalled 22.5. Using the trace files turned out to be more useful than we first expected, despite the performance gains of replaying pre-calculated files they also allowed us to test the performance of our traffic generator (setting all the sources on), cross check parameters as just stated as well as generating special sequences for analysing queue behaviour.

D.3 Traffic generator verification

We calculated the index of dispersion of intervals, or IDI (see Section III-D), also for the lab traffic generator. In Figure 12

we can see that the simulation and lab traffic generator produce similar types of traffic. The larger the observation time the more skewed the traffic is. One voice source is equal to about 18.1 also a value is given for a Poisson

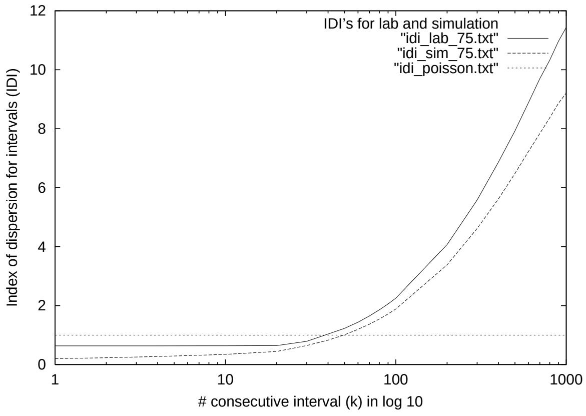

Fig. 12. IDI curves for superposition of 75 sources

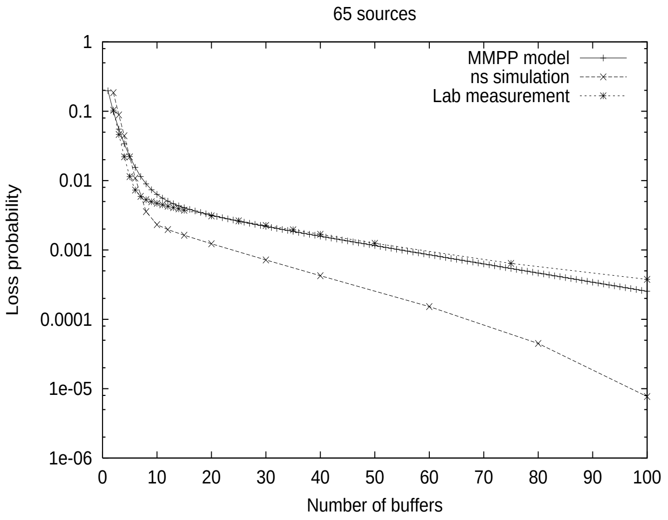

Fig. 13. 65 sources for model, NS and lab (log scale)

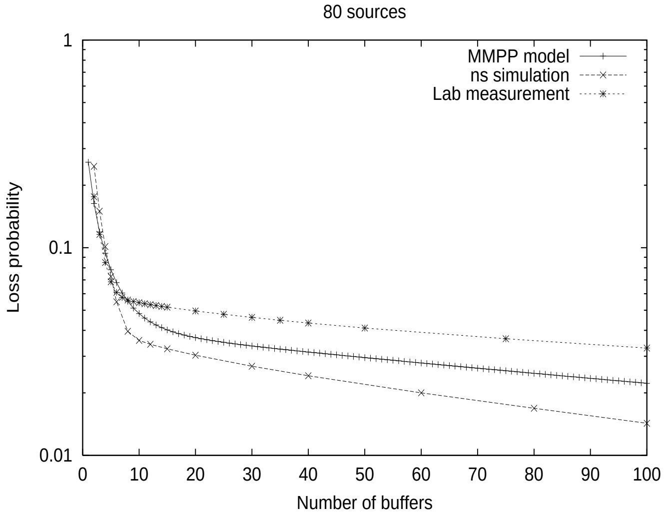

Fig. 14. 80 sources for model, NS and lab (log scale)

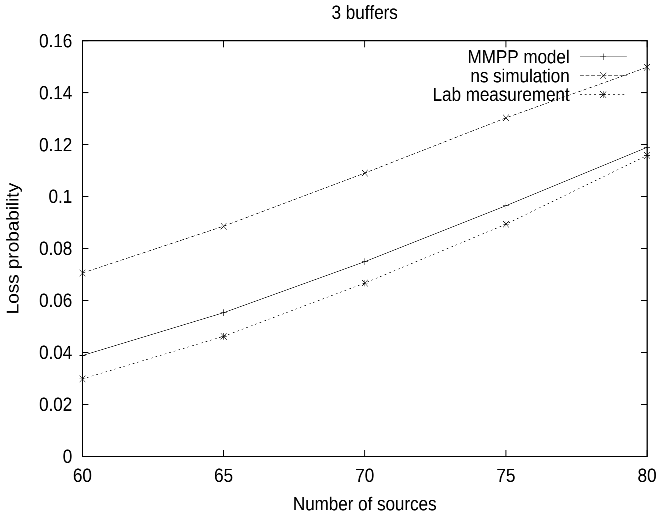

Fig. 15. Loss probability Vs buffers (3)

sources. The graphs show the result of a trace which was 10000 simulated seconds, resulting in 17.3 million packets for the lab and 16.3 for the simulation.

The traffic generator was also tested to ensure it (and the machine on which we run on) was capable of outputting packets as close to their deadlines as possible

VII. Results

In this section we present and discuss the results from the MMPP model, the NS simulations and the measurement in the lab network. Recall from Section IV that in all three cases we used the 32 kb/s ADPCM voice encoding with 16 ms packetization. This results in 64 bytes of voice payload in each packet and a total packet size of 104 bytes including the RTP, UDP and IP protocol headers.

Figures 13 and 14 show the packet loss probability as a function of the number of buffers on (y) log scales. We can see in these graphs that both the MMPP model results and the NS simulations in general compare well with the measurements in the lab. The exception is for very small buffer sizes and when the loss rate is small.

The MMPP model is most of the time closer to the lab measurements than the NS simulations are, which is an interesting result. The ns simulations consistently show the lowest loss rates for more than 7–8 buffers. We analysed the output from the traffic generators in NS and in the lab to try to come up with an explanation. We found that there is a small difference in mean total rate between the two that can explain the difference in loss rate.

The second set of graphs presented in Figures 15 to 18 plots the packet loss probability as a function of the number of voice sources for four different buffer lengths measured in packets. These buffer lengths correspond to a maximum queueing delay of 1.6, 2.7, 5.4 and 21.7 ms, respectively. We immediately see that the relationship be-

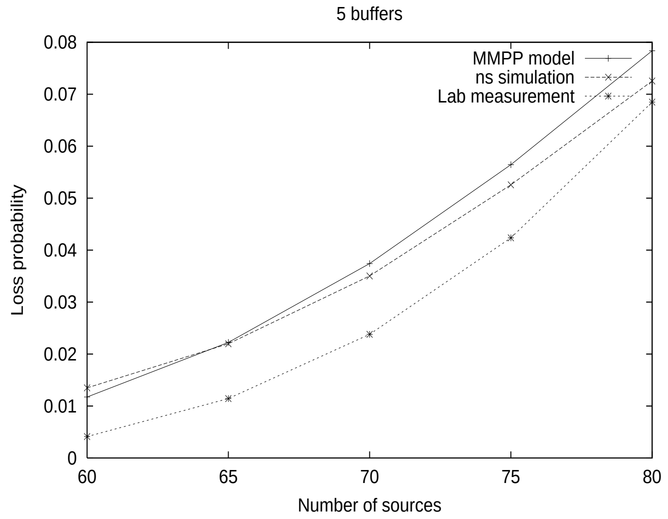

Fig. 16. Loss probability Vs buffers (5)

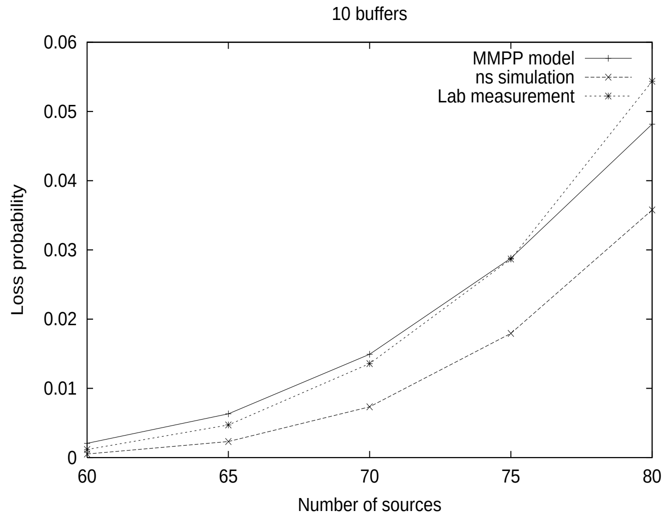

Fig. 17. Loss probability Vs buffers (10)

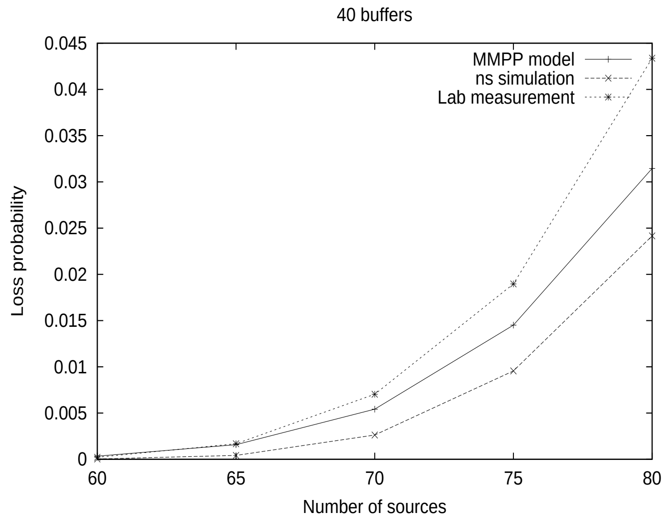

Fig. 18. Loss probability Vs buffers (40)

tween the number of sources and loss rate is close to linear for few buffers, but far from linear for many buffers. Visual observation suggests an exponential relationship. In the region above 10 buffers, the lab measurements often has the highest loss rate. Below about 10 buffers, the lab measurements have the lowest loss rate.

One interesting detail is that for very small buffers, the loss curve obtained in the NS simulation is shifted one buffer to the right in the plots. Even though we have gone to great lengths in ensuring that the three environments have identical properties, there are nevertheless subtle differences that can explain discrepancies like this.

One obvious difference in the models we used is that the bandwidth offered by Dummynet is not exactly the same as in ns. Using netperf we found there to be about a 3% difference between what netperf and dummynet report as their measured and configured bandwidths respectively. Perhaps more subtle and not so obvious is the amount of buffering in the system, in NS we simply state the buffer size in packets (between 2 and 100). In a real system this is much harder to calculate as buffers exist in many places in the system, for example in the queue between the Ethernet driver and `ip_input()` routine on the input side. Ethernet cards can also buffer packets on the output side. This is the default configuration as most Ethernet cards are used on host systems where this is not an issue. Nevertheless, the buffering in a real system is probably larger than the simulation and maybe account for differences in the systems under comparison.

VIII. Conclusions and future work

We have studied the packet loss behaviour when a number of homogeneous voice sources are multiplexed onto a bottleneck link. The goal is to find an accurate mathematical model which can be used to dimension the link.

We have implemented a mathematical model based on a Markov modulated Poisson process (MMPP) in Matlab. The model was compared with both simulations using NS and measurements in a lab environment. The comparison shows that the model in general predicts the loss rate well. The exceptions are for small loss rates in some cases. An interesting result is that most of the time the model predicts the loss rate better than the simulations in ns.

This result once more proves that the only way to reliably verify a model is to make measurements of a real system. We found that the relationship between the load and loss rate is close to linear for few buffers (around three), but looks exponential for many (10 and above) buffers.

The general conclusion is that the MMPP-based model is well suited for predicting loss rates for superpositioned voice sources in a system with limited buffer space. The mathematical model is an important tool for conveniently dimensioning network links. The lab environment is constrained to physical limits as well as finite resources where the model is clearly not. Running a lab experiment consumes resources and time a lab experiment takes on average 12 hours to complete. For each number of sources and each buffer size the experiment is re-started which is one reason why we use Dummynet, we can change the buffer sizes *without* re-booting the router. The simulation typically takes 2 hours whereas the model consumes only about 10 minutes as well as considerably less physical resources[3].

There are a number of further work items that we are currently addressing. The maximum delay is bounded by the buffer length in the system studied in this paper, but what is the resulting mean delay? We are experimenting with higher bandwidth links. One challenge is to accurately generate enough sources. The next step is to measure a system which has multiple traffic classes in the style of diffserv [11]. How does different queue scheduling algorithms affect the dimensioning of traffic classes? Can the MMPP model presented in this paper be used to describe the loss and delay properties of a traffic class?

The ongoing work can be found at a web page[4] which has information about current experiments as well as data which was not directly relevant for this paper.

Acknowledgements

The authors would like to acknowledge the indispensable work of Henrik Abrahamsson in helping us calculating the IDI values presented in this paper. We would like thank Thiemo Voigt for his help in setting up Dummynet and adding extra debug statements to make our loss calculations considerably simpler. Finally we would like to thank Telia AB for their financial support in the early phases of this work.

References

[1] Anders Andersson. Capacity study of statistical multiplexing for ip telephony. Technical Report T2000:03, SICS – Swedish Institute of Computer Science, January 2000.

[2] D. Anick, Debasis Mitra, and M. M. Sondhi. Stochastic theory of a data-handling system with multiple sources. *Bell System Technical Journal*, 61(8):1871–1894, October 1982.

[3] Andrea Baiocchi, Nicola Blefari-Melazzi, and Aldo Roveri. Buffer dimensioning criteria for an ATM multiplexer loaded with homogeneous on-off sources. In J. W. Cohen and Charles D. Pack, editors, *Queueing, Performance and Control in ATM — Proceedings of the Workshop at the 13th International Teletraffic Congress (ITC)*, pages 13–18, Copenhagen, Denmark, June 1991. North-Holland. Volume 15 of the North Holland Studies in Telecommunication.

[4] Andrea Baiocchi, Nicola Blefari Melazzi, Marco Listanti, Aldo Roveri, and Roberto Winkler. Loss performance analysis of an ATM multiplexer loaded with high-speed on-off sources. *IEEE Journal on Selected Areas in Communications*, 9(3):388–393, April 1991.

[5] Kevin Fall and Kannan Varadhan. ns: Notes and documentation. Technical report, Berkeley University, 1998. Technical Report.

[6] Allan Gut. *An Intermediate Course in Probability*. Springer-Verlag, New York, 1995.

[7] Harry Heffes and David M. Lucantoni. A Markov modulated characterization of packetized voice and data traffic and related statistical multiplexer performance. *IEEE Journal on Selected Areas in Communications*, SAC-4(6):856–867, September 1986.

[8] International Telecommunication Union (ITU). Transmission systems and media, general recommendation on the transmission quality for an entire international telephone connection; one-way transmission time. Recommendation G.114, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, March 1993.

[9] Steven McCanne and Van Jacobson. A BSD packet filter: A new architecture for user-level packet capture. In *Proc. of Usenix Winter Conference*, pages 259–269, San Diego, California, January 1993. Usenix.

[10] Ramesh Nagarajan, James F. Kurose, and Don Towsley. Approximation techniques for computing packet loss in finite-buffered voice multiplexers. *IEEE Journal on Selected Areas in Communications*, 9(3):368–377, April 1991.

[11] Kathie Nichols, Van Jacobson, and Lixia Zhang. A two-bit differentiated services architecture for the internet. Internet draft, Bay Networks, LBNL and UCLA, November 1997.

[12] L. Rizzo. Dummynet: A simple approach to the evaluation of network protocols. *Computer Communications Review*, 27(1):31–41, January 1997.

[13] Kotikalapudi Sriram and Ward Whitt. Characterizing superposition arrival processes in packet multiplexers for voice and data. *IEEE Journal on Selected Areas in Communications*, SAC-4(6):833–846, September 1986.

[14] Zheng Sun. Capacity study of statistical multiplexing for ip telephony. Technical Report LiTH-MAT-EX-98-12, Department of Mathematics, Linkoping University, December 1998.

[15] Roger C. F. Tucker. Accurate method for analysis of a packet-speech multiplexer with limited delay. *IEEE Transactions on Communications*, COM-36(4):479–483, April 1988.

[3] These values were derived from an Athlon 600 Mhz PC with FreeBSD, a Fast SCSI-3 disk and 128 MB of RAM.

[4] http://www.sics.se/~ianm/Experiment/experiment.html