Modelling and Analysis of a Transport Multicast Protocol

Alexandre Brandwajn [(1)][1*] and Serge Fdida [(2)]

UCSC-CRL-96-20

[(1)] University of California, Santa Cruz, 225 Applied Science Building, Santa Cruz, California, Tel. +1 408 459 4023, alexb@ce.ucsc.edu
[(2)] Laboratoire MASI-CNRS, Université P.&M. Curie, 4 place Jussieu, 75005 Paris, France
Tel. +33-1-44-27-30-58, Fax. +33-1-44-27-62-86, email: fdida@masi.ibp.fr

Contact person: Serge Fdida

[1*] This work was done while the first author was on sabbatical at MASI

Abstract

Point-to-multipoint communication is becoming a major need to handle multimedia applications and cooperative work. Multicast protocols are being designed in order to extend reliable transfer of information to multipoint configurations. Such protocols have to cope with several issues such as implosion of control packets that limits scalability or error management mechanisms to provide a reliable delivery of information to the end user. Most of these protocols are sensitive to the network environments and the parameters used to control the exchange of informations. It is therefore mandatory to properly size these parameters in order to operate the protocol efficiently. Performance analysis is then required. We consider a multicast algorithm akin to the bucket algorithm proposed in XTP3.6. We develop a queueing model that allows us to represent the key features of the proposed protocol, and assist in properly sizing its parameters. We discuss the model accuracy, and analyze the results as a function of various network environments.

I- Introduction

High speed networks, based on several technologies, are now under development with speeds projected to reach into Gbps. This dramatic increase in network bandwidth represents a tremendous opportunity to develop new services. In particular, multimedia, distributed computing and cooperative work appear as promising applications for the near future. Important new functionalities are required in order to satisfy the requirements of these emerging applications. The multicast facility, for instance, is expected to be widely used in groupware, distributed computing and multimedia applications. In this paper, we focus on a transport multicast protocol. Our goal is to analyze the performance of selected key aspects of such a mechanism, and to provide a first set of guidelines for the sizing of main parameters of the protocol in order to achieve a target Quality of Service (QoS).

The concept of multicast communications has received much recent attention. Service, protocols [Cha84, Bir91, Gar91, Ngo91, Raj92] and mechanisms [Agh94,Pin94] have been addressed for various network environments [Car94a, Dee94, Rob95]. One can point the work carried out in standardization committees (IETF, ANSI, ISO, ATM, etc. [Dee89, Coh91, Mou92, Coc92, ISO93, MPC95] and projects CIO [Mil93, Mat93], Berkom [Del93], Cesame [Ann94] for the definition of a multicast taxonomy, service and protocol. A transport layer multicast shall provide a simultaneous transfer of the same TPDU to a number of peer transport layer entities. In providing such a multicast service, the transport layer relies on the multicast capabilities provided by the network layer if any. Multicasting can be achieved at different levels. It is often natural at the MAC layer, also provided at the AAL layer (without error recovery), and at the network layer [Dee89]. When applied at the transport layer, functionalities to be used rely on the multicasting facilities provided at the lower layers, namely, either MAC, AAL or IP layer. Transport layer multicasting will be more easy to design if the

underlying service provide an efficient multicasting facility (for instance MAC or IP layer); if such a facility is not provided, other problems have to be solved such as packet replication, implosion, error recovery, addressing, group management.

We consider a communication situation where a single data stream is sent from a transmitter to a set of receivers. This mechanism, named multicast, is designed to operate on different classes of networks. We restrict ourselves to a multicast facility, akin to XTP3.6 [PEI92, San92, PEI94], designed to work on top of a Local Area Network that handles the broadcast property. Moreover, the transmitter does not require any knowledge of group membership. An extension of this protocol to ATM based networks and large group management is presented in [Fdi96]. Other approaches are presented in [Car94b], [Flo95].

The basic purpose of the multicast protocol under study is to provide a reliable error control mechanism for multicasting operation. The transmitter sends control and data packets to a set of K receivers (known as a multicast group) that responds by sending back control packets (such as acknowledgements). The multicast system under consideration is shown in Figure 1.

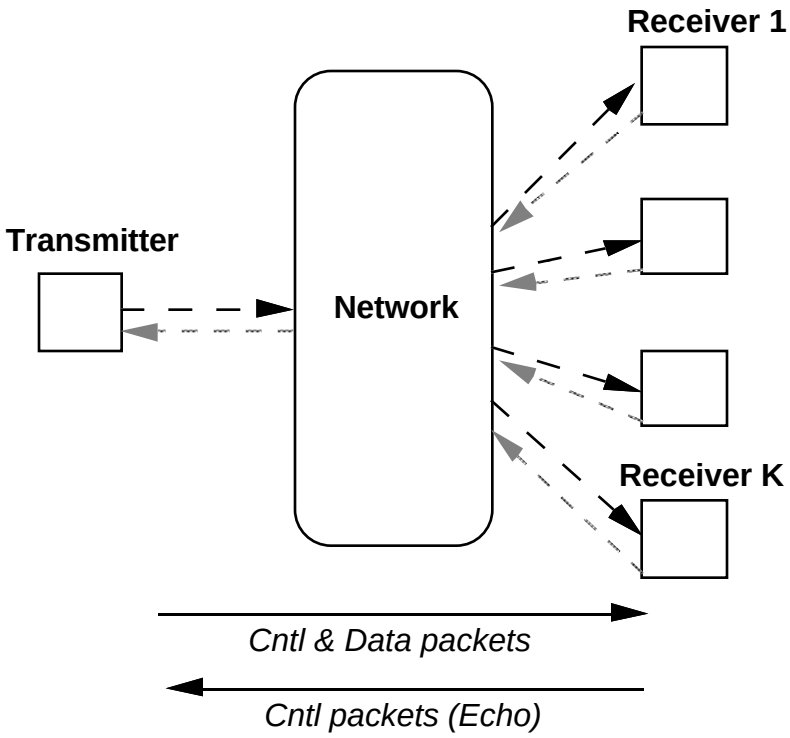

Figure 1: The multicast system

The paper is organized as follows. Section 2 describes the basic operation of the multicast protocol under consideration, akin to XTP3.6. Section 3 presents the model of the system. Its solution is obtained under memoryless assumptions, using a fixed-point iterative procedure. Section 4 is devoted to the analysis and sizing of the key parameters of the system. With the help of the results of discrete-event simulations, we investigate also the accuracy of our model. Finally, Section 5 concludes the paper and outlines further studies.

II- System description

We now briefly describe the basic mechanisms of the multicast operation on which we base our analysis. The protocol works roughly as follows. Periodically, every Δ time units, the sender sends control packets identified by a set of parameters including a sequence number (sync). A receiver responds to such a control packet by a packet of its own identified by the same sequence number (echo=sync), and which carries additional information (composite information). Thus, the sequence numbers ensure that a response control packet is associated with the correct request control packet.

The bucket algorithm is used to manage the multicast error control procedure. When the sender transmits a control packet identified by a given sync value, it expects to receive a response to this control packet within a specified time window (Tw). Reply packets received within this window are accumulated and analyzed so as to extract the most conservative values.

The system has at its disposal a fixed number of buckets, and a bucket is associated with every control packet issued by the transmitter in order to hold the information of the corresponding reply packets. Thus, the information contained in a bucket serves to identify packets that have been correctly received at a given point in time.

The protocol recognizes as errors the following three conditions: a packet is received with errors, a packet is lost, or the reception of the reply packet is delayed beyond the lifetime of the corresponding bucket. When a bucket has to be freed, the information it contains allows the sender to determine which packets, if any, should be retransmitted. We assume here that the retransmission procedure is Go-Back-N (all packets starting form the first one in error are retransmitted). As response packets can be lost or delayed, this multicast mechanism is only statistically reliable.

The bucket is a data structure containing fields of interest in the management of the error recovery procedure such as dseq and rseq. rseq is the sequence number of the last packet "correctly" received by the receiver protocol entity, and dseq is the number of the last packet passed to the receiver service user entity.

The periodicity with which control packets are generated is referred to as the control period Δ. Clearly, a consistency relationship must exist between the number of buckets (Tw/Δ), the round trip delay and the control period. The lifetime of a bucket extends from the moment a bucket is created to the moment it has to be recycled to make space available for another control packet. To ensure correct operation, this lifetime should be greater than the round trip delay. The greater the lifetime of a bucket, the lower the number of retransmissions due to delayed reply packets. On the other hand, because the sender's input buffer has a finite capacity and is updated when a bucket is released, a longer bucket lifetime

will tend to increase the probability of packet loss at the sender due to buffer overflow.

The lifetime of a bucket is related to the number of buckets in the system. A larger number of buckets allows to increase the frequency of sampling (i.e., decrease the control period). More frequent sampling reduces the number of packets to be retransmitted in case of error. This has a positive effect on the throughput of the system. However, more frequent sampling increases the overhead both in terms of the amount of information transmitted and in terms of processing required (which appears to be unacceptable for moderate to large group size). This tends to have a negative effect on system throughput. Thus, there are several tradeoffs involved, and the reliability achieved using this mechanism is a function of Tw, Δ, and the network parameters. It is the goal of this paper to provide a first level sizing of these protocol parameters.

The control packet operation is illustrated in Figure 2.

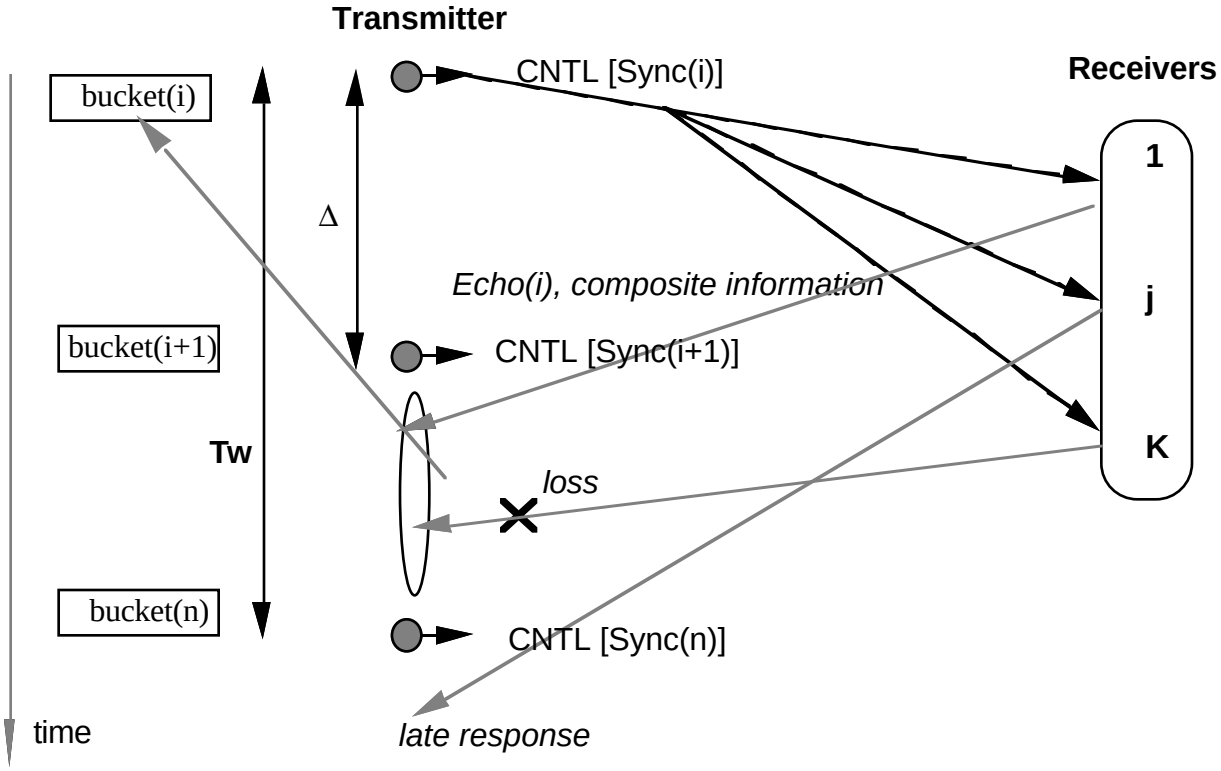

Figure 2: Control packet operation

III- Model of the system

We propose a model of the multicast protocol in which both the sender and the receivers are viewed as finite capacity queues. The network delay is represented by a delay server (infinite server queue). Packets can be lost both at the sender queue and at the receiver queues due to buffer overflow. We consider a system in which all receivers are statistically identical, so that we use a single queue to model the set of receivers. The resulting model is shown in Figure 3.

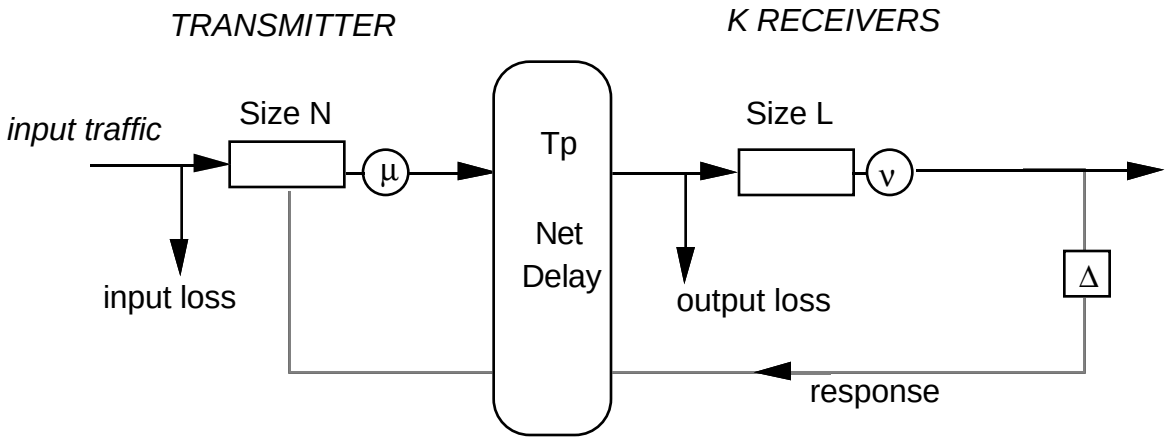

Figure 3: Model of the system

3.1- Assumptions and Notations

We consider a set of K receivers. Each receiver responds immediately to a control packet. As mentioned before, an error can occur either because of a transmission error over the network, a packet loss at the receiver entity, or simply because of a delayed arrival at the sender of the response control packet from a receiver (this latter condition is referred to as Delayed Echo Packets).

As a first step, we are interested in evaluating the throughput of the system as a function of input and output buffer sizes and the duration of the control period. We assume that Tw is sized according to the maximum network round-trip delay so that delayed Echo packets never occurs. In this case, the control period (Δ) is a key parameter, because it also fixes the number of buckets (Tw/Δ).

We use the following notations:

- referring to the input buffer, we call class 1 customers messages awaiting transmission. We call class 2 those messages which have been transmitted and stored waiting for an acknowledgement,
- T_p is the mean one way propagation delay over the network, and we let $\gamma=1/(2T_p)$,
- $p_{in}(n_1,n_2)$ is the probability that there are n_1 class 1 customers and n_2 class 2 customers in the input buffer Bin,
- $P_{in}(n)$ the probability that there are n customers (class 1 plus class 2) in the input buffer Bin,
- $p_{in,2}(n_2)$ probability that there are n_2 customers of class 2 in the input buffer Bin,
- $P_{out}(n)$ probability that there are n customers in the output buffer Bout,

For simplicity and tractability, in the sequel, we assume that service times at the sender and receiver are exponentially distributed with mean $1/\mu$ and $1/\nu$, respectively. Similarly, the propagation network round-trip delay and the control interval are assumed to be exponentially distributed with mean $1/\gamma$ and $1/\delta$

respectively. The impact of some of these distributional assumptions was studied using discrete event simulation. Section 4 addresses this subject.

3.2- Sender analysis

For our purpose here, we view the sender as an input buffer of size N. The input traffic is assumed to be Poisson with parameter λ. The packets sent are stored in the buffer until an acknowledgement is received allowing to delete from the buffer the number of packets corresponding to the indications contained in the acknowledgment. Our model accounts for the fact that several data packets can be acknowledged by a single response packet. If a transmission error occurs, the sender enters a retransmission procedure according to the response control packets received so far.

The proposed model of the input buffer is shown in Figure 4.

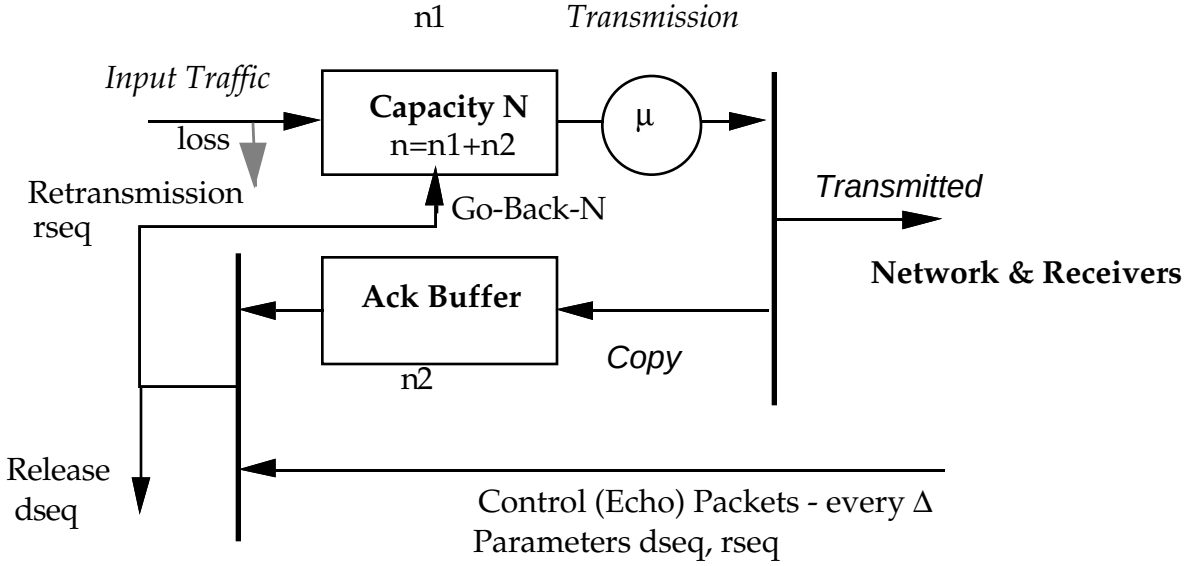

Figure 4. The input buffer model

The packets sent enter the input buffers if there is space available or are lost otherwise. They wait for transmission (population n_1) in the buffer. Once they are transmitted, a copy is kept in the input buffer until the corresponding acknowledgement is received (population n_2). When the corresponding bucket is recycled (after Tw), a group of packets is released from the input buffer (corresponding to the value dseq), and another group could be retransmitted according to the received value rseq.

Referring to the bucket recycling instant let:

$p_a(k \mid n_1, n_2)$ be the probability to free a group of k packets (k=0,…, n_2) from the input buffer knowing that there are n_1 packets waiting for transmission and n_2 for acknowledgement.

$p_b(m \mid n_1,n_2)$ be the probability to retransmit a group of m packets from the input buffer knowing that there are n_1 packets waiting for transmission and n_2 for acknowledgement.

We denote by N and L the capacity of the input and output buffers, respectively. $n_1 + n_2 \leq N$.

The following general balance equations are obtained for the input buffer model:

1- $n_2 < L$. In this situation, there are no retransmissions.

$$p_{in}(n_1,n_2)\,[\lambda+\mu+\delta^*(n_1,n_2)]$$

$$= p_{in}(n_1-1,n_2)\,\lambda + p_{in}(n_1+1,n_2-1)\,\mu \qquad \{\text{arrivals + transmission}\}$$

$$+ \delta \sum_{k=1}^{N-(n_1+n_2)} p_a(k \mid n_1,n_2+k)\, p_{in}(n_1,n_2+k) \qquad \{\text{release of k packets}\}$$

where $\delta^*(n_1,n_2) = \delta \sum_{k=1}^{n_2} p_a(k \mid n_1,n_2)$

2- $n_2 = L$. This is a particular state because it is entered upon retransmission due to buffer overflow of the output buffer (capacity L).

$$p_{in}(n_1,n_2=L)\,[\lambda+\mu+\delta \sum_{k=1}^{L} p_a(k \mid n_1, L)]$$

$$= p_{in}(n_1-1,L)\,\lambda + p_{in}(n_1+1,L-1)\,\mu \qquad \{\text{arrivals + transmission}\}$$

$$+ \delta \sum_{k=1}^{N-(n_1+L)} p_a(k \mid n_1,L+k)\, p_{in}(n_1,L+k) \qquad \{\text{release of k packets}\}$$

$$+ \delta \sum_{m=1}^{n_1} p_b(m \mid n_1-m,L+m)\, p_{in}(n_1-m,L+m) \qquad \{\text{retransmission of m packets}\}$$

3- $n_2 > L$.

$$p_{in}(n_1,n_2)\,[\lambda+\mu+\delta \sum_{k=1}^{n_2} p_a(k \mid n_1, n_2) + \delta \sum_{m=1}^{n_2-L} p_b(m \mid n_1, n_2)]$$

$$= p_{in}(n_1-1,n_2)\,\lambda + p_{in}(n_1+1,n_2-1)\,\mu \qquad \{\text{arrivals + transmission}\}$$

$$+ \delta \sum_{k=1}^{N-(n_1+n_2)} p_a(k \mid n_1,n_2+k)\, p_{in}(n_1,n_2+k) \qquad \{\text{release of k packets}\}$$

The conditional probabilities $p_a()$ and $p_b()$ are derived from models of other systems components as described hereafter and summarized in Section 3.6.

From the input buffer analysis, we obtain the input buffer probabilities and hence the traffic directed to the output buffer, taking into account losses that occur at the input buffer:

$$\lambda^* = \lambda\, [\, 1\text{-}P_{in}(N)]$$

3.3- Receiver analysis

The set of receivers is taken to comprise K statistically identical stations. The station generates answers to the CNTL packets issued by the sender. Answers are broadcast to the group (sender and receivers) and consist mainly of acknowledgement packets used to free the input buffer, and to initialize the retransmission procedure in case of error.

The receiver queue is viewed as an output buffer of capacity L, and its model is represented in Figure 5.

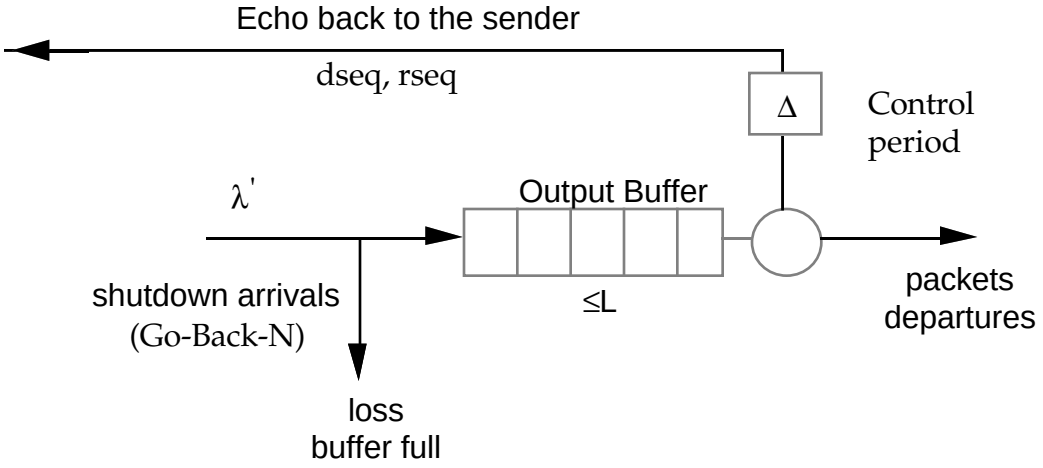

Figure 5. The output buffer : model of a receiver

λ' is the mean total rate of arrivals of data packets at the output buffer. It includes both first transmissions and retransmissions from sender to receivers. This traffic λ' enters the output buffer if free space is available or is lost otherwise (and will later generate retransmission of the lost packet and subsequent ones). The control procedure is modeled through the transmission back to the sender of the Echo packets. They are sent periodically every Δ and carry the control parameters dseq and rseq.

3.4- Analysis of the output buffer

We are now interested in the analysis of the output queue that represents the behavior of receiver's buffers. For a given receiver, the number of packets to be retransmitted corresponds to the set of packets between the first received in error (loss or transmission error) and the last packet stored in the input queue because the retransmission procedure is Go-Back-N. To simplify our analysis, we start by neglecting transmission errors, i.e., we assume that the only errors possible are those caused by output buffer overflow. According to this assumption, as long as the output buffer is not full, every single packet will be received correctly and acknowledged to the source. As soon as the output buffer becomes full, the protocol management is able to inform the source that the retransmission procedure must be attempted.

An important point is that during the period from the instant when the loss occurs to the moment when the first recovered packet arrives at the output queue, no packet will be admitted in the output buffer due to Go-Back-N. Hence, to model loss of packets due to buffer full condition, we shut down arrivals for a period of duration $1/\tau = 2T_p + 1/\nu + \Delta/2$, where T_p is the one-way network delay, $1/\nu$ is the average treatment time for a message at the receiver, and Δ is the duration of the control period. In order for the procedure to be efficient, the output loss will have to be kept very small, so that we assume that the input flow (initial packets plus retransmissions) is still Poisson with parameter λ'.

In our analysis of an output buffer, we consider that the output buffer can be in one of two states: state 1, referred to as Regular, prevails when no loss occurs, and state 2, referred to as the Shutdown state, occurs when an arriving data packet finds the buffer full. We denote the system state as p(n,i), n=0,…,L, and i=1,2, where n is the number of packets stored in the output queue and i is the current status (Regular or Shutdown). Also, we let $P_{out}(n)$ be the number of packets stored in the output queue.

To solve the equations for p(n,i), we observe that p(n,i) can be expressed as $p(n,i) = P_{out}(n)\, p(i \mid n)$, where $p(i \mid n)$ is the conditional probability that the output buffer is in state Regular or Shutdown given the number of packets.

$P_{out}(n)$ is then given by

$$P_{out}(n) = G \prod_{i=1}^{n} \frac{\tilde{\lambda}'(i-1)}{\nu}, \qquad n=1,\ldots, L$$

where G is a normalizing constant

with $\tilde{\lambda}'(i) = \lambda'[1-p(2 \mid i)], \qquad i=0,\ldots, L$

and we obtain the following recurrent solution for the conditional probabilities $p(i \mid n)$.

For n=L:

From the balance equation:

$$p(L, 2)\,(\tau+\nu) = p(L, 1)\,\lambda'$$

we get, $p(2 \mid L)\,(\tau+\nu) = p(1 \mid L)\,\lambda'$

so that, $p(2 \mid L)\,(\tau+\nu) = [1-p(2 \mid L)]\,\lambda'$

$$p(2 \mid L) = \frac{\lambda'}{\lambda'+\nu+\tau}$$

then for n=L-1, …, 1:

$$p(2 \mid n)\,(\tau+\nu) = p(2 \mid n+1)\,\nu\ \frac{\lambda'\,[1-p(2 \mid n)]}{\nu}$$

$$p(2 \mid n)\,(\tau+\nu) = \lambda'\,p(2 \mid n+1) - \lambda'\,[p(2 \mid n+1)\,p(2 \mid n)]$$

$$p(2 \mid n)\,\{\tau+\nu+\lambda'\,p(2 \mid n+1)\} = \lambda'\,p(2 \mid n+1)$$

$$p(2 \mid n) = \frac{\lambda'\,p(2 \mid n+1)}{\lambda'\,p(2 \mid n+1)+\nu+\tau}$$

and finally for n=0:

$$p(2 \mid 0) = \frac{\lambda'\,p(2 \mid 1)}{\lambda'\,p(2 \mid 1)+\tau}$$

The solution of the output buffer enables to compute $P_{out}(L)$, which is the loss probability that fires retransmissions. The number of packets to be retransmitted is equal to the number stored in input buffer minus the capacity of the output buffer. We assume, as necessary for an efficient operation of the system that, the input buffer capacity N is larger than the output buffer capacity L.

3.5- Additional models

Two additional models are developed in order to capture
1- the bulk size distribution of packets that are positively acknowledged and thus released from the input buffer every control period. We refer to this model as the acknowledgement model,
2- the auxiliary model is used to capture the distribution of outstanding packets that are either in transit within the network or in the output queue knowing that n_2 packets are awaiting acknowledgement in the input buffer.

3.5.1- The acknowledgement model

The acknowledgement model is designed to yield the bulk size distribution of the positively acknowledged packets knowing that n_2 packets are waiting for an acknowledgement. We start by the case of a single receiver. The corresponding model is shown in figure 6.

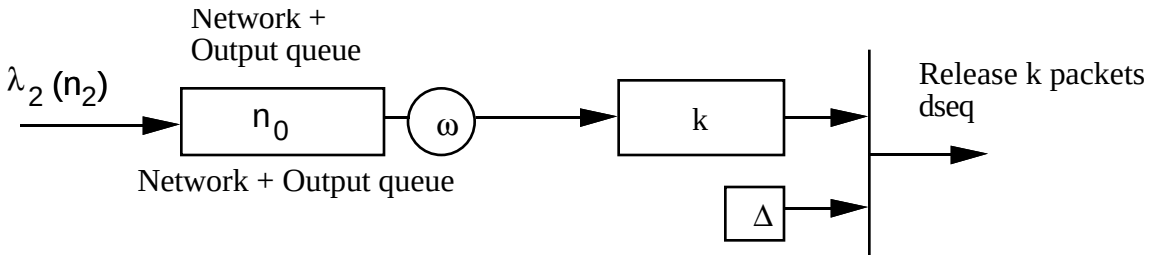

Figure 6. The acknowledgement model

We use a memoryless state dependent process with rate $\lambda_2(n_2)$ for arrivals. The first queue represents the time from the instant a packet is passed to the network by the sender to the moment a corresponding Echo packet will reach the input queue. We use the throughput of the output buffer queue as the service rate of the corresponding "server":

$$\omega(n_0)=\nu\ [1-p_{out}(n_s=0 \mid n_0)]$$

and $\lambda_2(n_2) = \text{Prob}(n_1>0 \mid n_2)\ \mu$

with $n_0 + k = n_2$

where k is the number of packets to be released.

We use the following notations:
$p_{out}(n_s=0 \mid n_0)$ is the conditional probability that the output queue is empty given the total number of packets in network and output queue, and is obtained using the auxiliary model described in 3.5.2.

$\text{Prob}(n_1>0 \mid n_2)$ is the conditional probability that the sender has packets to send given that n_2 await acknowledgement, and it can be derived from the solution of the input queue.

Let $p(n_0 , k)$ be the joint probability that there are a total of n_0 packets in the network and the output queue and k packets can be acknowledged. We readily obtain the following balance equations:

1- k>0

$$p(n_0, k)\ [\lambda_2(n_0+k) + \delta + \omega(n_0)]$$

$$= p(n_0-1, k)\ \lambda_2(n_0+k-1) + p(n_0+1, k-1)\ \omega(n_0) \quad \text{for } n_0>0$$

$$p(0, k)\ [\lambda_2(k) + \delta] = p(1, k-1)\ \omega(n_0) \quad \text{for } n_0=0$$

2- k=0

$$p(n_0, 0)\,[\lambda_2(n_0) + \omega(n_0)] = p(n_0\text{-}1, 0)\,\lambda_2(n_0\text{-}1) + \sum_{k=1}^{N-n_0} p(n_0, k)\delta \qquad \text{for } n_0 > 0$$

and

$$p(0, 0)\,\lambda_2(0) = \sum_{k=1}^{N} p(0, k)\delta$$

Again, we find it convenient to resort the conditional probabilities through the identity $p(n_0, k) = p(n_0)\,p(k \mid n_0)$. Substituting this relationship into balance equations for $p(n_0, k)$, and using the fact that $p(n_0)$ can be expressed as

$$p(n_0) = H \prod_{i=1}^{n_0} \frac{\lambda_0(i\text{-}1)}{\omega(n_0)}$$

where $\lambda_0(n_0) = \sum_{k=0}^{N-n_0-1} \text{Prob}(k \mid n_0)\,\lambda_2(n_0+k)$

and H is a normalizing constant.

we get:

1- k=0

$$p(0 \mid 0) = \frac{\delta}{\lambda_2(0) + \delta}$$

$$p(0 \mid n_0) = \frac{\delta + p(0 \mid n_0\text{-}1) + \dfrac{\lambda_2(n_0\text{-}1)\,\omega(n_0)}{\lambda_0(n_0\text{-}1)}}{\lambda_2(0) + \omega(n_0) + \delta}$$

2- k>0

$$p(k \mid n_0)\,[\lambda_2(n_0+k) + \delta + \omega(n_0)]$$

$$= p(k \mid n_0\text{-}1)\,\lambda_2(n_0+k\text{-}1)\,\frac{\omega(n_0)}{\lambda_0(n_0\text{-}1)} + p(k\text{-}1 \mid n_0+1)\,\lambda_0(n_0)$$

and $\sum_{k=0}^{N-n_0} \text{Prob}(k \mid n_0) = 1$

It is not difficult to solve these equations using a fixed point iteration. Thus, the acknowledgement model yields the probability $p(n_0 , k)$ in the case of a single receiver. Its use and application to multiple receivers are discussed in section 3.6.

3.5.2- The auxiliary model

This model is used to approximately compute the probability to find n_s packets in the output buffer for a given load of n_0 packets in the system. We denote by $p_{out}(n_s|n_0)$ this probability. The model is shown in figure 7.

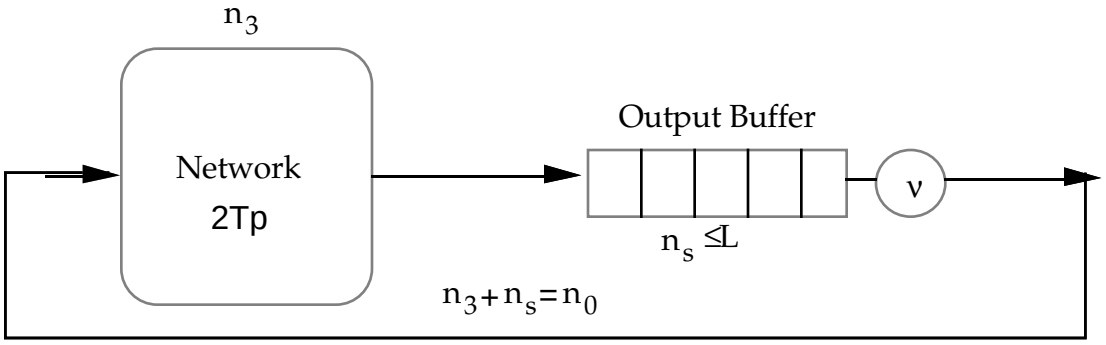

Figure 7. The auxiliary model

We compute the probability that the output buffer overflow given the number of packets waiting for an acknowledgement as the probability that the output queue is at capacity but in its Regular state (not yet Shutdown), and a packet arrives from the network during a service time of the output queue.

This yields: $p(\text{overflow} \mid n_2) =$

$$\sum_{n_0=L}^{n_2} p_{out}(n_s=L \mid n_0)\; p(n_0 \mid n_2)\; p(\text{one arrival during } 1/\nu)\; p(i=1 \mid L)$$

The first term of the right hand side of the equation is obtained from the auxiliary model, $p(n_0 \mid n_2)$ from the acknowledgement model, and the last term from the output model.

3.5.3- Workload from other stations

The traffic patterns are described by the source traffic. However, additional traffic can be received by any receiver or the sender from other connections. In a first step, we would like to incorporate in our model the feedback control traffic from the receivers due to the Echo packets. Let T_{echo} be the time to process an Echo packet by the sender. This control traffic will consume a percentage α of the sender's processing capability. We represent this effect as an elongated service time, i.e. reduced processing rate at the server:

$$\mu_r = \mu[1-\alpha] \text{ , where } \alpha = K\frac{T_{echo}}{\Delta}$$

3.6- Solution procedure

As described above, we have decomposed the system under consideration into four models: the input buffer at the sender, the output buffer at the receiver, the acknowledgement process to derive the distribution of the number of messages released each control period, and the auxiliary model to account for network transit delays. It is apparent from our discussion that these models depend on one another in the sense that results of one model determine the values of input parameters for another model. This, naturally suggests an iterative approach.

First, we solve the auxiliary model outside the fixed point iteration. Its solution produces $p_{out}(n_s|n_0)$, the conditional probability that there are n_s packets in the output queue given that there are n_0 packets in the network and output queue.

Thus, starting with the offered traffic λ as the initial value for λ', the rate of message arrivals to the output queue, we consider the models in the following order.

1- the output buffer model

Its solution produce, the conditional probability $p(i=1|n=L)$ that the output buffer is in its regular state given that it is at capacity, and θ, the number of packets processed per time unit. Additional performance measures, such as the distribution of the number of packets at the receiver or the mean sojourn time, can also be computed.

2- the acknowledgement model

Its solution produces the probability $p(n_0, k)$ that there are n_0 packets in the network and output queue, and k packets to be acknowledged. This allows us to compute $p(k|n_2)$, the conditional probability that k packets are acknowledged given the total awaiting acknowledgements ($n_2=n_0+k$), as well as the conditional $p(n_0|n_2)$. Using $p(n_0|n_2)$ together with $p_{out}(n_s|n_0)$ we evaluate the probability of output buffer overflow given n_2: $p(overflow|n_2)$. $p(k|n_2)$ is used as an approximation for $p_a(k|n_1,n_2)$, and $p(overflow|n_2)$ is used for $p_b(m|n_1,n_2)$. Note that in both cases we neglect the dependence on n_1, i.e., we treat n_2 as the dominant condition.

The acknowledgement model of section 3.5.1 considers a single receiver. With multiple receivers, the protocol will retain the most conservative value of the number of packets acknowledged. The K receivers in our set are all subject to the same input stream and, therefore, can not be treated as K independent units. Based on comparisons with discrete event simulation, we find that, when the number of receivers exceeds 4 or 5, the distribution of the number of packets acknowledged is quite accurately represented by considering the convolution of just two identical distributions $p(k_i,n_2)$. The $p(k_i,n_2)$ come from the solution of

the acknowledged model with a single receiver, and we combine them so as to get the most conservative case: $p\{k=l|n_2\}$ = Prob $\{k_1 \geq l$ and $k_2 \geq l|n_2\}$.
It is the result of this transformation that we use as the probability that k packets are acknowledged given that a total of n_2 are awaiting acknowledgment. Note that, in some sense, it is as if the two worst receivers were masking all the other ones since our acknowledgment model corresponds to a receiver with no "advance" over other receivers in the set. For a small number of receivers, such as 2 our 3, our results tend to be conservative.

Both $p(k|n_2)$ and $p(\text{overflow}|n_2)$ are used as parameters in the input buffer model. To solve the acknowledgement model, we need $p_{out}(n_s=0|n_0)$ from the auxiliary model, and $p(n_1>0|n_2)$ from the input buffer model.

3- the input buffer model

Its solution produces $p(n_1,n_2)$, the joint probability that there are n_1 packets to be retransmitted and n_2 packets awaiting acknowledgement. Hence, we get $p(n_1|n_2)$ and the admitted traffic $\lambda^* = \lambda\,[1-P_{in}(n_1+n_2=N)]$.

In the fixed point approximation, we use the value of λ^* obtained from the analysis of the input buffer to determine the value of λ', the rate of packet arrivals to the output queue, so that the throughput θ of the latter equals the admitted traffic at the sender. A straightforward adjustment

$$\lambda'^{i+1} = \lambda'^{i} \cdot \frac{\lambda^{*i}}{\theta}, \; i=1,2, \ldots$$

where the superscript denotes the iteration number, turns out to work well in practice.

We do not have a formal proof of convergence. Experimental evidence indicates that the proposed iteration converges within just a few iterations: typically on the order of 10 for a relative difference between consecutive values of λ^* less than 10^{-5}.

IV- Results and Analysis

This section is devoted to the analysis of both the method accuracy and results obtained for many network configurations.

The approximation approach described above enable us to compute system parameters such as throughput, delay, buffer size distribution, loss probabilities, distribution of the number of packets released/retransmitted per control period, etc. To assess its accuracy, we compared our method with simulation. In the simulation, we considered both constant and exponential distribution for the parameter Δ in order to check the influence of this distributional assumption. We ran a number of simulations for many network configurations and compared the key parameters. The results indicate that our approach is a robust method able to

take into account many difficult features of the system. Having gained confidence in the method's performance, we used it in order to study the system behaviour as a function of different network scenarios.

We used the following parameters to check the method accuracy against simulation:
$P_{in}(i)$ is the probability to find i (i=1,…,N) customers in the input queue,
$P_{out}(j)$ is the probability to find j (j=1,…,L) customers in the output queue,
$P(n_2)$, n_2=1,…,N, is the distribution of the number of packets awaiting for acknowledgment in the input buffer,
$P_{lib}(k)$, k=1,…, N-L, is the distribution of the number of packets released from the input buffer per control period Δ,
Total Delay is the delay experienced by a packet to be correctly delivered to the end-user.

Table 1 compares results obtained with simulation (for a selected receiver) and the model for the following parameters:
K= 10, N=15, L=6, Tp=1ms, Δ=100ms, Techo=1ms, λ=50pack/s, μ=200pack/s, ν=100pack/s.

Recall that K is the number of receivers, N is the capacity of the input buffer at the sender, L denotes the capacity of the output buffer at each of the receivers, Δ is the duration of the control period, Techo is the time to process an echo packet at the sender, λ is the average rate of arrival of packets to the sender, 1/μ is the mean time to process a packet at the sender, and 1/ν is the mean time to process a packet by the receiver (exclusive of queueing in the buffer).

Parameters	Analytic	Simulation (Δ=Exp)	Simulation (Δ=Cst)
$P_{in}(0)$	0.0153	0.017	0.012
$P_{in}(N)$	0.118	0.148	0.012
$P_{out}(0)$	0.515	0.531	0.458
$P_{out}(N)$	0.006	0.008	0.012
$P(n_2=0)$	0.02	0.023	0.017
$P(n_2=1)$	0.053	0.044	0.049
$P(n_2=2)$	0.083	0.065	0.083
$P(n_2=3)$	0.098	0.078	0.108
$P_{lib}(0)$	0.184	0.165	0.0045
$P_{lib}(1)$	0.143	0.138	0.026
$P_{lib}(2)$	0.116	0.118	0.069
Tot. Delay(ms)	31.84	35.24±0.52	35.22±0.38

Table 2 presents results obtained with simulation and the model for the following parameters:

K= 10, N=15, L=6, Tp=20ms, Δ=100ms, Techo=1ms, λ=50pack/s, μ=200pack/s, ν=100pack/s.

Parameters	Analytic	Simulation (Δ=Exp)	Simulation (Δ=Cst)
$P_{in}(0)$	0.0015	0.0078	0.0015
$P_{in}(N)$	0.174	0.182	0.034
$P_{out}(0)$	0.546	0.550	0.467
$P_{out}(N)$	0.004	0.007	0.009
$P(n_2=0)$	0.002	0.01	0.002
$P(n_2=1)$	0.007	0.02	0.012
$P(n_2=2)$	0.021	0.033	0.03
$P(n_2=3)$	0.046	0.054	0.055
$P_{lib}(0)$	0.195	0.204	0.0042
$P_{lib}(1)$	0.148	0.142	0.026
$P_{lib}(2)$	0.118	0.114	0.074
Tot. Delay(ms)	51.01	53.79±0.78	53.53±0.47

Table 3 shows the results obtained with simulation and the model for the following parameters:

K= 10, N=15, L=6, Tp=10ms, Δ=100ms, Techo=1ms, λ=25pack/s, μ=200pack/s, ν=100pack/s.

Parameters	Analytic	Simulation (Δ=Exp)	Simulation (Δ=Cst)
$P_{in}(0)$	0.015	0.059	0.075
$P_{in}(N)$	0.014	0.010	0.0
$P_{out}(0)$	0.729	0.73	0.723
$P_{out}(N)$	0.0002	0.0002	0.0003
$P(n_2=0)$	0.018	0.069	0.087
$P(n_2=1)$	0.11	0.144	0.185
$P(n_2=2)$	0.178	0.17	0.222
$P(n_2=3)$	0.175	0.155	0.193
$P_{lib}(0)$	0.288	0.275	0.079
$P_{lib}(1)$	0.204	0.211	0.199
$P_{lib}(2)$	0.145	0.150	0.26
Tot. Delay (ms)	31.6	31.7±0.39	31.77±0.24

Although we cannot present all the results here, we see that they are generally in good agreements with the simulation results obtained for an exponentially

distributed control period (Δ). Simulation results for a constant control period differ from the analytical model results mainly for the computation of the distribution of packets released every Δ. However, the key parameters are still within an acceptable relative error even for worst case configurations. We found that the relative error tends to be larger for smaller values of Δ, and larger values of Tp. Given the very simple network model used, the latter effect is not surprising. The increase in relative error as Δ becomes small is not fully understood.

The analysis of the protocol behavior is presented in figures 8 to 12. The influence of the control period and the propagation delay on the offered traffic and the acknowledgement delay (time for a packet to get released from the input buffer) is analyzed. We can see that there exists a value of Δ that minimizes the loss and therefore optimizes the throughput and the delay. We notice in figure 8 and 9 that the influence of the propagation delay is negligible. Figure 10 shows the influence of the input load on the throughput. There always exists an optimum value for Δ. It is worth to mention that the optimum value for Δ seems to depend little on the offered load. Figure 11 shows the influence of the number of receivers on the delay and throughput that leads to the well-known implosion problem.

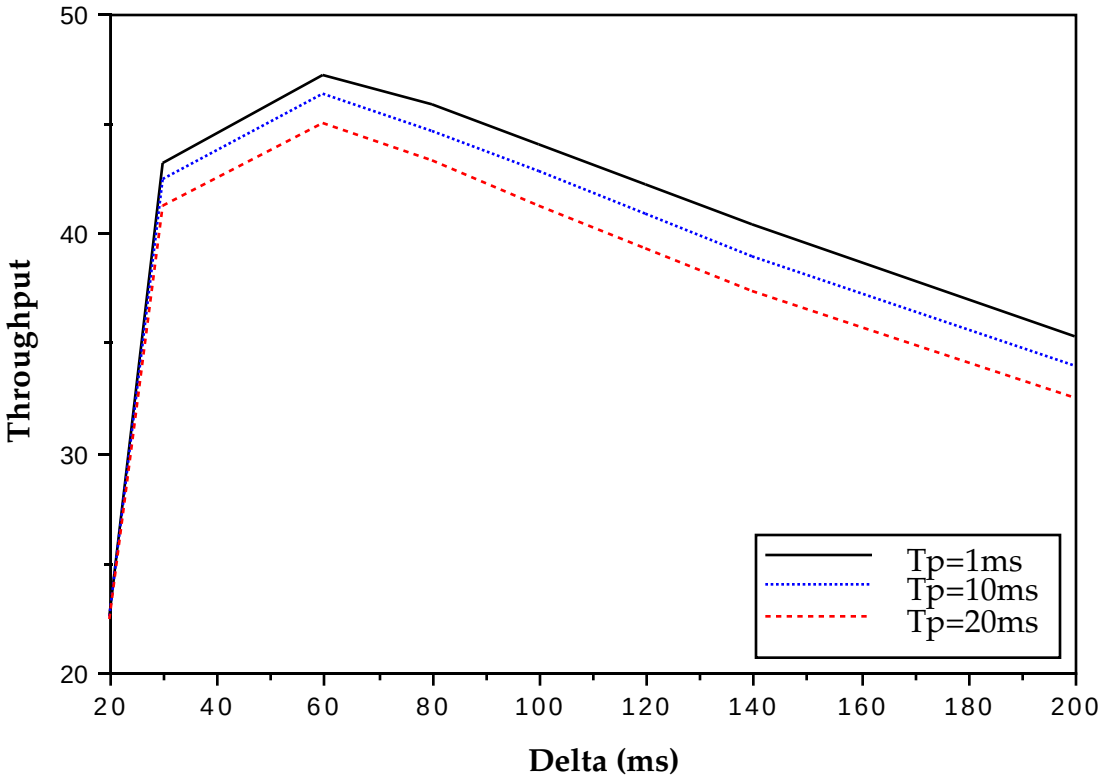

Figure 8. Influence of Δ and Tp on the Throughput (pack./s)
(K= 10, N=15, L=6, Techo=1ms, λ=50pack/s, μ=200pack/s, ν=100pack/s.)

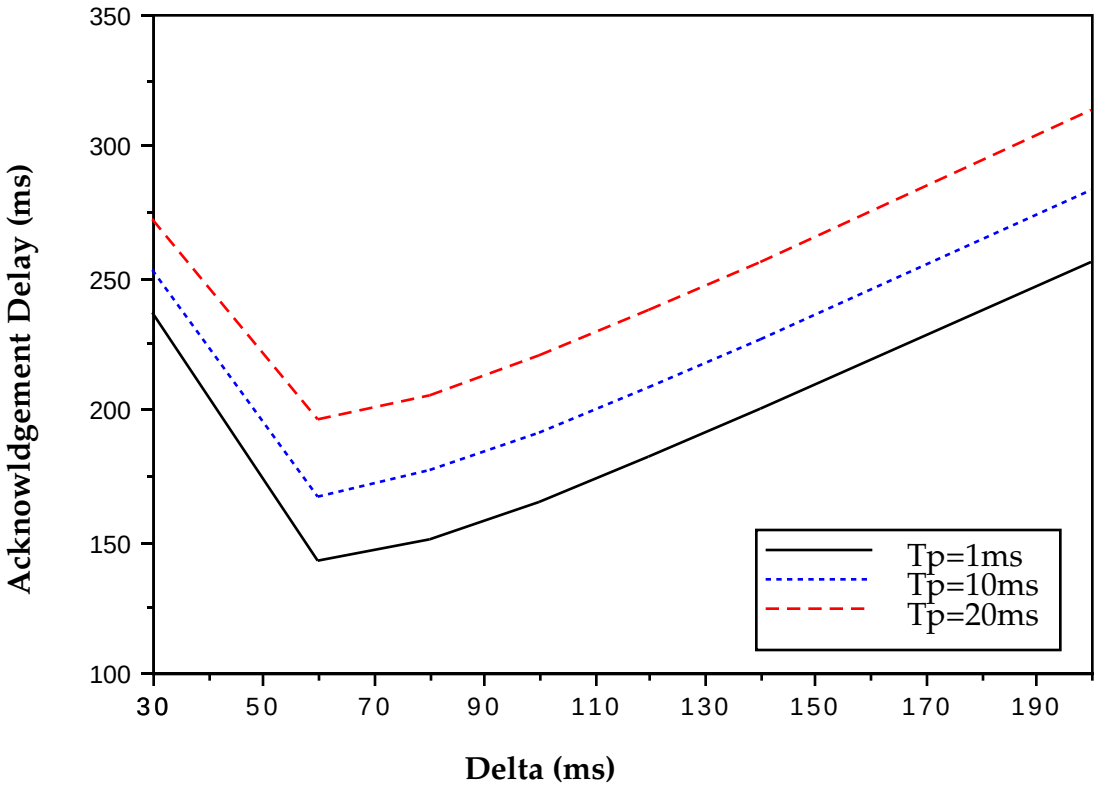

Figure 9. Influence of Δ and Tp on the Acknowledgement Delay (ms) (K= 10, N=15, L=6, Techo=1ms, λ=50pack/s, μ=200pack/s, ν=100pack/s.)

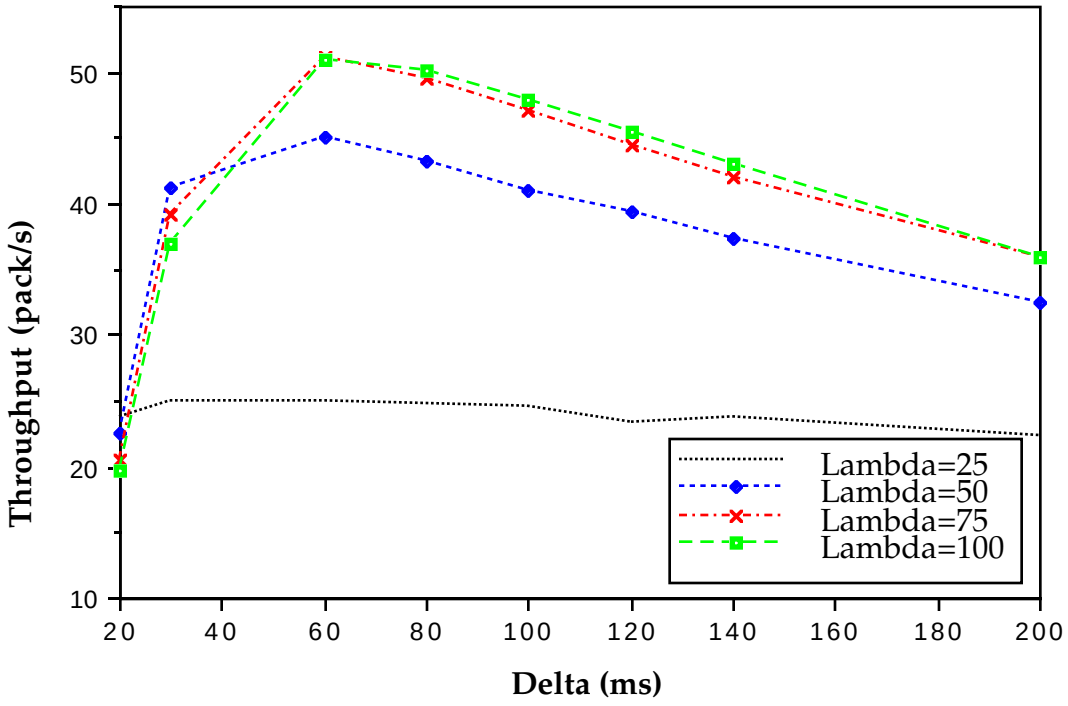

Figure 10. Influence of the load (packets/s) and Δ (ms) on the Throughput (K= 10, N=15, L=6, Techo=1ms, Tp=20ms, λ=50pack/s, μ=200pack/s, ν=100pack/s.)

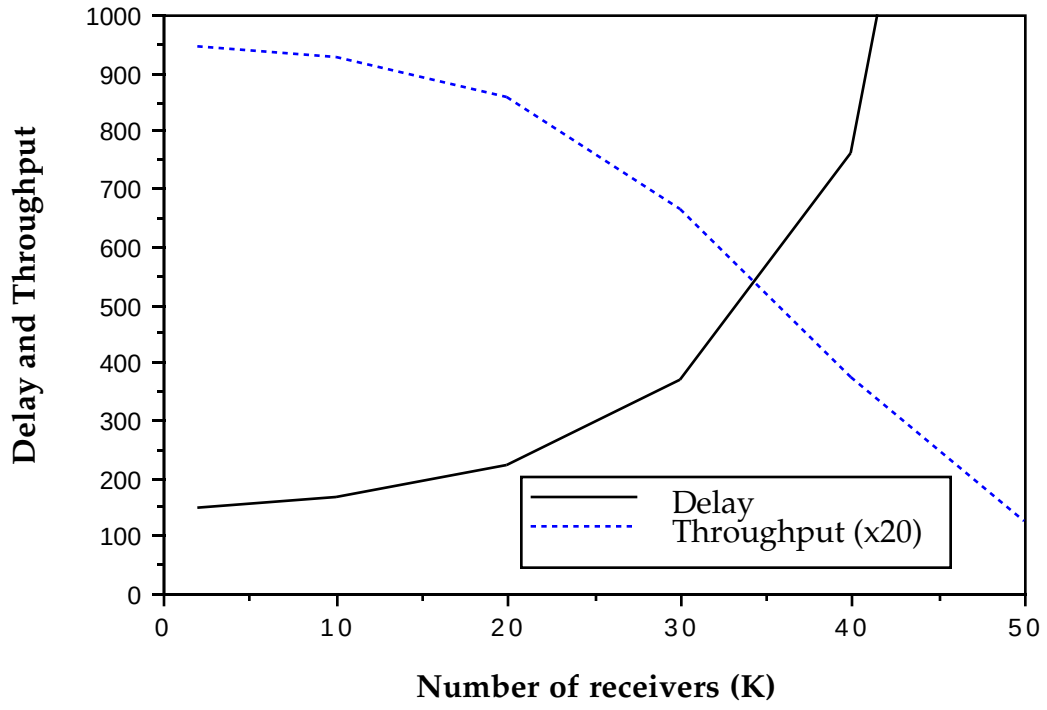

Figure 11. Influence of the number of receivers on the Throughput (pack./s) and Acknowledgement Delay (ms)
(Delta=60ms, N=15, L=6, Techo=1ms, Tp=10ms, λ=50pack/s, μ=200pack/s, ν=100pack/s.)

In our model, the number of buckets could be derived from the RTT (Round-Trip-Time) divided by the control period Δ. However, the computation of RTT is a key issue in a real environment. In fact, RTT varies from time to time according to the network and traffic changes, making the number of buckets variable in time. This is also the case in our system. Another solution is to compute an approximate value for RTT and then select a value for Δ. In such a situation, the number of buckets is kept constant during the life of the conversation. Figure 12 illustrate just such a case. We see that there exists an optimum number of buckets which depends on the RTT value chosen. A solution is to build an algorithm with either a dynamic number of buckets to react to changes in RTT (keep Δ constant) or, adjust Δ to keep the number of buckets fixed.

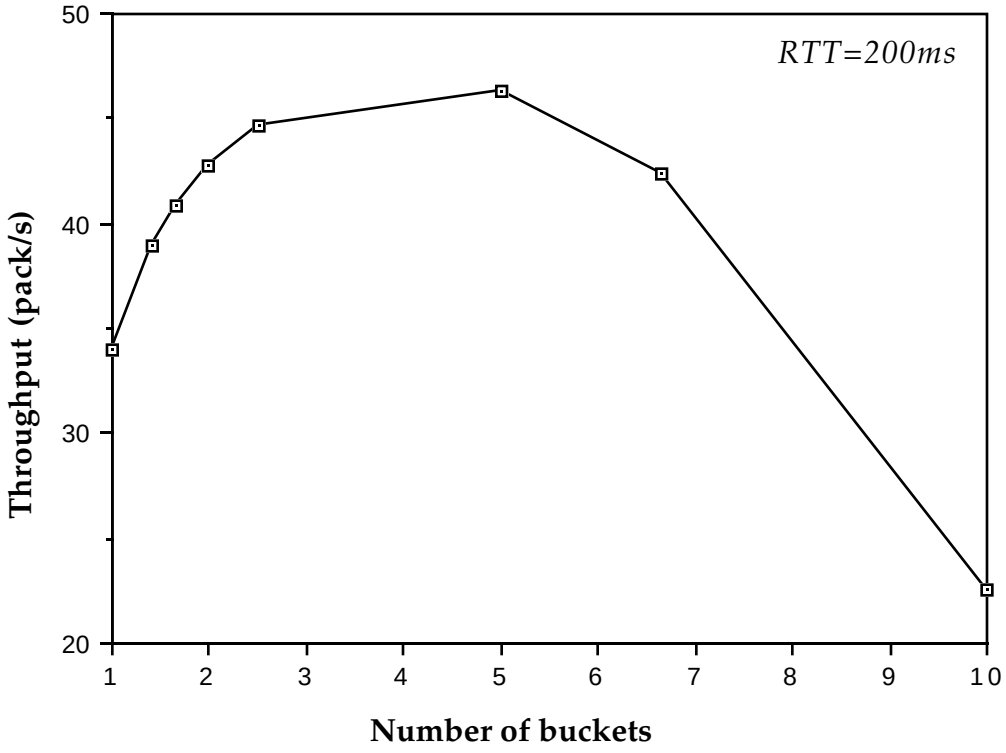

Figure 12. Influence of the number of buckets on the throughput (RTT chosen=200ms)
(K= 10, N=15, L=6, Techo=1ms, λ=50pack/s, μ=200pack/s, ν=100pack/s.)

V- Conclusion

Among other services, multimedia cooperative applications require a multicast capability with various levels of reliability. The efficient provision of a reliable multicast service is difficult and needs to be carefully addressed and analyzed as a function of the network, application and key parameters of the protocol. In this paper, we considered a statistically reliable multicast protocol akin to XTP3.6. A model able to handle several complex features of this protocol was derived and compared against simulation results. The model allowed us to analyze and size selected protocol parameters, and to show that the system throughput is optimized for some values of the number of buckets used in the algorithm.

V- References

[Ann94] Annals of Telecommunications, The Cesame Project, Tome 49, N°5-6, pp. 217-356, May-June 1994.[30]

[Bha94] Bhagwat P., Mishra P., Tripathi S.K., "Effect of Topology on Performance of Reliable Multicast Communication", Infocom94, pp.602-609, 1994.

[Bir91] Birman K. , Schiper A. and Stephenson P., "Lightweight Causal and Atomic Group Multicast", ACM Transactions on Computer Systems, vol. 9, no. 3, pp. 272-314, August 1991.

[Car94a] Carle G., "Adaptation Layer and Group Communication Server for Reliable Multipoint Services in ATM Networks", in Steinmetz (Ed), Multimedia: Advanced Teleservices and High-Speed Communication Architectures, Springer, pp. 124-138, 1994.

[Cha84] Chang J. and Maxemchuk N.F. , "Reliable Broadcast Protocols", ACM Transactions on Computer Systems, vol. 2, no. 3, pp. 251-273, August 1984.

[Coc92] Cocquet P., Diot C., "Enhanced Transport Service", Proposed Contribution to ISO/IEC JTC1 SC6/WG4, June 1992.

[Coh91] Cohn M., "High Speed Transport Protocol (HSTP) Specification", Contribution to ISO/IEC JTC1 SC6/WG4 on the High Speed Transport Protocol, September 1991.

[Dee89] Deering S., "Host Extension for IP Multicasting", RFC 1112, August 1989.

[Dee94] Deering S. et al., "An Architecture for Wide-Area Multicast Routing", SIGCOMM'94, London, pp.126-135, august 1994.

[Del93] Delgrossi L., Sandvoss J. (ed.), "The BERKOM-II Multimedia Transport System (MMT), Version 3.0", August 1993.

[Fdi96] S. Fdida, "Multimedia Transport Protocol and Multicast Communication", in: High-Speed Networking for Multimedia Applications, W. Effelsberg, O. Spaniol, A. Danthine, D. Ferrari (eds.), Kluwer Academic Publishers, Boston/Dordrecht/London, 1996.

[Flo95] S. Floyd et al., "A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing", SIGCOMM'95, pp.342-356, 1995.

[Gar91] Garcia-Molina H. and Spauster A., "Ordered and Reliable Multicast Communication", ACM Transactions on Computer Systems, vol. 9, no. 3, pp. 242-272, August 1991.[5]

[ISO93] ISO/IEC, JTC1/SC6/WG4, Draft Multicast Taxonomy of Multicast Operation, 10.31.1993.

[Mat93] Mathy L., Leduc G., Bonaventure O., Danthine A., "A Group Communication Framework", CIO RACE Project 2060, R2060/ULg/CIO/IN/P/005, December 1993.

[Mil93] Miloucheva I., "Specification of Enhanced Protocol Facilities for Multicast and Broadcast", CIO RACE Project 2060, R2060/TUB/CIO/DS/P/003/b1, October 1993.

[Mou92] Moulton J. , Proposed USA Contribution to SC6 on Multicast Transport Protocol, July 1992.

[MPC95] Multi-Peer Communication Architecture, ISO/IEC Draft 7498-5, SG7-SC21, 1995.

[Ngo91] Ngoh L.H., "Multicast Support for Group Communications", Computer Networks and ISDN Systems, vol. 22, pp. 165-178, 1991.

[Ott94] J. Ott, C. Borman, "Multicasting the ITU MCS: Integrating Point-to-multipoint and Multicast Transport", Singapore ICCS, Elsevier, pp. 1013-1017, 1994.

[Pin94] S. Pingali, D. Towsley, J. Kurose, "Comparison of Sender-initiated and Receiver initiated Reliable Multicast Protocols", Performance Evaluation Review, Vol.22, N.1, pp. 221-230, 1994.

[PEI92] Protocol Engines Inc., "XTP Protocol Definition", Version 3.6, January 1992.

[PEI94] Protocol Engines Inc., "XTP Protocol Definition", Version 4.0, 1994.

[Raj92] Rajagopalan B., "Reliability and Scaling Issues in Multicast Communications", Sigcomm'92, pp188-198, 1992.

[Rob95] Roberts L.G., "Point-to-Multipoint ABR Operation", ATM Forum /95-0834, August 1995.

[San92] Santoso H., Fdida S., "Transport Layer Multicast:An Enhancement for XTP bucket algorithm", 4th IFIP High Performance Networking (hpn'92), Liège, Belgique, December 1992.