Performance Evaluation of Sequential Decoding System for UDP-Based Systems for Wireless Multimedia Networks

Khalid Darabkh and Ramazan Aygün, *Member, IEEE*

ABSTRACT—Sequential decoders are convolutional channel decoders that are characterized by having variable decoding complexity in changing channel conditions. They are widely used in wireless packet-switching networks and mobile communications. The major difference of multimedia networks from traditional networks is the deadline to display or (play) the arriving packets. Especially, this is more obvious in real-time video streaming applications. Real-time video streaming applications prefer User Datagram Protocol (UDP) for video packet transfers. The operation of sequential decoding in this paper is applied on wireless multimedia networks. Buffers are usually required by the sequential decoder to store packets if there is an empty space. Geometric/Pareto/1/N queue model is used to describe the behavior of the sequential decoders with finite buffers. In this paper, we present an analytical study on the finite buffer behaviors of the sequential decoding system and provide closed form expression for the blocking probability due to limited buffer capacity, system throughput, and average buffer occupancy. We present also a simulation study to show the average buffer occupancy.

KEYWORDS—Sequential decoder, average buffer occupancy, queuing analysis, blocking probability, system throughput.

I. INTRODUCTION

The estimation of buffering requirements in wireless networks is significant due the cost of buffers and energy consumption based on the buffer size. However, convolutional coding [17] is one of the most important techniques used for channel coding. Its main goal is to reduce the probability of erroneous transmission over noisy channel. Figure 1 show the block diagram of the coding mechanism. Mainly there are two important decoding algorithms for convolutional codes, the Maximum-Likehood Decoding (Viterbi's algorithm), and Sequential decoding.

Viterbi decoding [16] was developed by Andrew J. Viterbi, a founder of Qualcomm Corporation. It has a fixed decoding time. It is well suited to hardware decoder implementation. Its computational and storage requirements grow exponentially as a function of the constraint length (2^L), where L is the number of stages in the shift register that is used at the encoder side, and are very attractive for constraint length L<10. To achieve very low error probabilities, longer constraint lengths are required. Thus, sequential decoding becomes attractive. Convolutional coding [20] with Viterbi decoding has been the predominant FEC (Forward Error Correction) technique used in space communications, particularly in satellite communication networks, such as VSAT (very small aperture terminal) networks.

Sequential decoding was first introduced by Wozencraft for the decoding of convolutional codes [2] [18] [19]. The sequential decoding complexity increases linearly rather than exponentially. It has a variable decoding time. Sequential decoding [7] [8] [9] is well-known for its computational complexity being independent of the code constraint length. Sequential decoding can achieve a desired bit error probability when a sufficiently large constraint length is taken for the convolutional code. The decoding complexity of a sequential decoder becomes dependent to the noise level [6]. These specific characteristics make the sequential decoding useful in particular applications

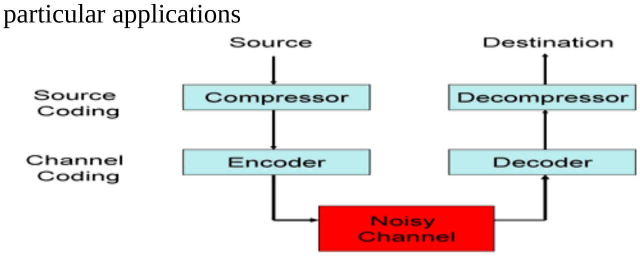

Figure 1: Block diagram of coding

The operation of sequential decoding in this paper is considered in wireless multimedia networks. The Discrete-time Markov section gives more details about that. The major advantage of the sequential decoding algorithm is its amenability to practical implementations. In a practical decoding system, however, buffers are required to absorb the variable processing delays of the sequential decoders. If the buffer overflows, all those incoming packets that cannot find a space to reside in will be discarded (dropped), these packets are considered to be lost. Figure 2 shows the sequential decoding system.

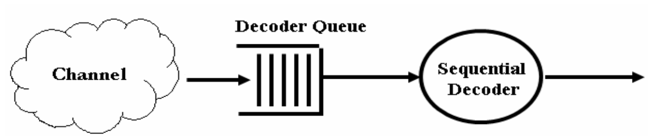

Figure 2. Sequential decoder with a finite buffer.

Khalid Darabkh is with Computer Engineering Department, University of Alabama in Huntsville, Huntsville, AL 35899, USA (e-mail: darabkk @eng.uah.edu).
Ramazan Aygün is with Computer Science Department, University of Alabama in Huntsville, Huntsville, 35899, USA (e-mail: raygun @cs.uah.edu).

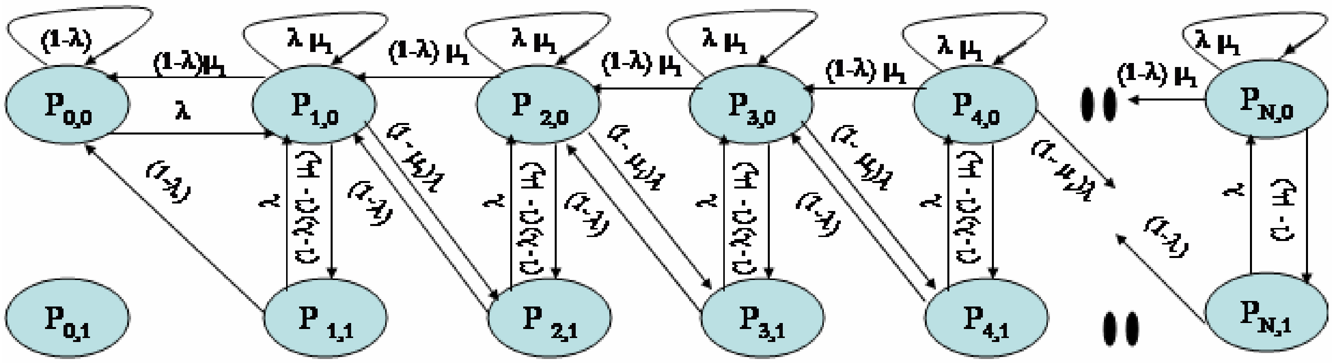

Figure 3: Probability state transitions for $\text{Geometric}/\text{Pareto}/1/N$ queue with two slots for decoding

It has been reported both analytically and experimentally [5] [4] [7] [12] that the tail of the distribution of the sequential decoding time behaves according to the Pareto law. The decoding time can be formalized as a function of the channel SNR (signal-to-noise ratio) [11] [15]. When the SNR is low, it means the channel is noisy, so the decoding time needs to be large, on the other hand, if the SNR is low, it means the channel is good (not noisy), so the decoding time needs to be small. The decoding time is often limited by a timeout to avoid blocking of the decoder. We assume that packets from the channel arrive to a finite buffer of the decoder according to a Bernoulli process, and that the decoding time of the sequential decoder follows the Pareto distribution parameterized by the channel SNR. We model the sequential decoding system using a discrete time Semi-Markov model and our queueing analysis yields a closed form expression for the blocking probability due to limited buffer capacity, system throughput, and average buffer occupancy. We present also a simulation study to show the average buffer occupancy. The agreement between analytical and simulation studies is shown. Analytical and simulation results are presented for several different channel conditions, packet arrival rates, and buffer capacities.

This paper is organized as follows. The following section provides the Discrete Time Markov Model, which forms the basis of modeling our system. Section 3 provides the theoretical analysis of our approach. The simulation setup and results are provided in Section 4. The last section concludes our paper.

II. DISCRETE TIME MARKOV MODEL

The sequential decoding system is analyzed using discrete-time Markov model. This model is used for wireless multimedia networking systems. The time axis is portioned into slots of equal length where each slot corresponds to exactly equal to the transmission time. We assume that all the incoming packets are assumed to be equally length. This is the case if we send Internet packets (typically of the size of 1Kbytes) over a wireless link (where packets have the size of around 300 bytes). Thus, the decoder can receive at most one new packet during a slot. The buffer is finite. Therefore, if a packet arrive where there is no space in the buffer (buffer overflow), then this packet and all the following packets will be dropped. Thus, these packets are considered to be lost. However, the new packets arrive at the decoder from the channel according to Bernoulli process. A slot carries an arriving packet with probability λ and it is idle (no transmission) with probability 1- λ. The service time follows the Pareto distribution with a parameter β which is a function of SNR. The buffer size is assumed to be at least one room. We make another assumption that the decoding time of a packet is in chunks of length equal to the slot size. That is, the decoder can start and stop decoding only at the end of a slot. This assumption replaces the continuous distribution function of the decoding time by a staircase function that is a pessimistic approximation of the decoding time. This approximation yields an upper bound on the number in the queue [4] [10].

Therefore, we assume in our analytical and simulation studies that any packet is just allowed up to two slots for decoding. If a packet requires j slots for decoding ($j \leq 2$), it leaves the system at the end of the j^{th} slot after the beginning of its decoding, and the decoding of a new packet starts (if there is a new packet in the decoder's buffer) at the beginning of the following slot. If a packet's decoding needs more than two slots, the decoder stops that packet's decoding after two slots. This packet cannot be decoded and thus a decoding failure results and this packet is considered lost. The queue is analyzed according to a method known as supplementary variables [4].

III. THEORETICAL ANALYSIS

A. *Decoding Time Distribution*

We use the notations that are mentioned in prior research in [4] [15]: c_k denotes the probability of decoding being completed in exactly k slots and μ_k denotes the conditional probability that decoding is completed in k slots given that the decoding is longer than (k – 1) slots. Then the conditional probability μ_k is given by

$$\mu_j = \frac{c_j}{1 - F_{j-1}}\,, \tag{1}$$

where $F_j = \sum_{i=1}^{j} c_i$ is the cumulative distribution function (CDF) of the decoding time.

It can be shown that

$$\prod_{i=1}^{j} (1 - \mu_i) = 1 - F_j \tag{2}$$

The decoding time of sequential decoders has the Pareto distribution:

$$P_F(\tau) = \Pr\{t > \tau\} = \left(\frac{\tau}{\tau_0}\right)^{-\beta}, \tag{3}$$

where τ_0 is the decoding time for which the probability is 1, i.e., the minimum time it takes the decoder to decode a packet, and β is called the Pareto parameter and it is a function of the SNR of the channel.

From (1) and (2), we have

$$F_j = 1 - P_F(jT_r) \quad 1 \le j \le T, \quad T_r \text{ is the slot duration} \tag{4}$$

and

$$c_j = F_j - F_{j-1} \quad 1 \le j \le T\,. \tag{5}$$

B. Analysis

The state of the queue can be represented by the pair (n, t), where n is the number of packets in the buffer including the packet being decoded. Here we assume the buffer is of finite length, hence ($0 \le n \le N$). t is the number of slots the decoder has already spent on the packet that is currently being decoded. If it takes more than 2 slots for the packet to be completely decoded, then we have a decoding failure. The packet is thus considered lost.

Let $P_{n,t}$ be the probability that the decoder's buffer contains n packets, including the one being decoded by the decoder, which is in the t^{th} slot of decoding. Figure 3 shows the probability state transitions for Geometric/Pareto/1/N queue model. The summation of all the outgoing links (probabilities) from each state must be equal to one. **T**he steady-state transitions equations are:

$$P_{0,0} = (1-\lambda)P_{0,0} + \mu_1(1-\lambda)P_{1,0} + (1-\lambda)P_{1,1}, \tag{6}$$

$$p_{0,1} = 0, \tag{7}$$

$$p_{1,0} = \mu_1\left[\lambda p_{1,0} + (1-\lambda)p_{2,0}\right] + \lambda p_{1,1} + (1-\lambda)p_{2,1} + \lambda p_{0,0}\,, \tag{8}$$

$$p_{1,1} = (1-\lambda)(1-\mu_1)p_{1,0} \tag{9}$$

$$p_{n,0} = \mu_1\left[\lambda p_{n,0} + (1-\lambda)p_{n+1,0}\right] + \lambda p_{n,1} + (1-\lambda)p_{n+1,1}, \quad 2 \le n \le N-1 \tag{10}$$

$$p_{n,1} = (1-\mu_1)\left[(1-\lambda)p_{n,0} + \lambda p_{n-1,0},\right] \quad 2 \le n \le N-1 \tag{11}$$

$$p_{N,0} = \mu_1\left[\lambda p_{N,0}\right] + \lambda p_{N,1}. \tag{12}$$

$$p_{N,1} = (1-\mu_1)\left[p_{N,0} + \lambda p_{N-1,0}\right] \tag{13}$$

We define the following probability generating function:

$$P_j(z) = \sum_{n=1}^{N} p_{n,j} z^n, \quad 0 \le j \le 1 \tag{14}$$

And the probability generating function of the number of packets in the buffer

$$P(z) = p_{0,0} + \sum_{j=0}^{1} P_j(z). \tag{15}$$

From equations (6), (8),(10), and (12) we have:

$$p_0(z) = \sum_{j=1}^{2}\mu_j\left[\lambda\sum_{n=1}^{N-1} p_{n,j-1}z^n + (1-\lambda)\sum_{n=1}^{N-1} p_{n+1,j-1}z^n\right] + \lambda z p_{0,0} \tag{16}$$

$$p_0(z) = \sum_{j=1}^{2}\mu_j\left[\lambda(p_{j-1}(z) - p_{N,j-1}z^N) + \frac{(1-\lambda)}{z}\left[P_{j-1}(z) - P_{1,j-1}z\right]\right] + \lambda z p_{0,0} \tag{17}$$

$$p_0(z) = \sum_{j=1}^{2}\mu_j\lambda p_{j-1}(z) - \sum_{j=1}^{2}\mu_j\lambda p_{N,j-1}z^N + \sum_{j=1}^{2}\mu_j\frac{(1-\lambda)}{z}P_{j-1}(z) - \sum_{j=1}^{2}\mu_j\frac{(1-\lambda)}{z}P_{1,j-1}z + \lambda z p_{0,0} \tag{18}$$

$$p_0(z) = \sum_{j=1}^{2}\mu_j\lambda p_{j-1}(z) - P_{N,0}z^N + \sum_{j=1}^{2}\mu_j\frac{(1-\lambda)}{z}P_{j-1}(z) - \lambda p_{0,0} + \lambda z p_{0,0} \tag{19}$$

$$p_0(z) = \frac{1}{z}\sum_{j=1}^{2}\mu_j(1-\lambda+\lambda z)p_{j-1}(z) - P_{N,0}z^N + \lambda(z-1)p_{0,0} \tag{20}$$

Let $f(z) = (1-\lambda+\lambda z)$ (21)

From equations (9), (11), and (13) we have:

$$P_1(z) = p_{1,1}z + \sum_{n=2}^{N} p_{n,1}z^n \tag{22}$$

$$P_1(z) = (1-\mu_1)\left[(1-\lambda)\sum_{n=1}^{N-1} p_{n,j-1}z^n + \lambda z\sum_{n=1}^{N-1} p_{n-1,0}z^{n-1}\right] \tag{23}$$

$$P_1(z) = (1-\mu_1)\left[\begin{array}{l}(1-\lambda)\left(P_0(z) - P_{N,0}z^N\right) \\ +\lambda z\left(P_0(z) - P_{N-1,0}z^{N-1} - P_{N,0}z^N\right)\end{array}\right] \tag{24}$$

$$P_1(z) = (1-\mu_1)\left[f(z)P_0(z) - P_{N,0}z^N f(z) - \lambda P_{N-1,0}z^N\right] \tag{25}$$

$$\mu_1 = \frac{c_1}{1-F_0} = c_1 = F_1 \tag{26}$$

Let $y = F_1\lambda - 1$ (27)

Let $x = 1 - F_1$ (28)

From (12) and (27) we have: $p_{N,1} = \dfrac{-y\, p_{N,0}}{\lambda}$ (29)

From (13), (27), (28), and (29) we have:

$$p_{N-1,0} = \frac{p_{N,0}\left[\dfrac{-y}{x\lambda} - 1\right]}{\lambda} \tag{30}$$

From (25) and (30):

$$P_1(z) = x\left[f(z)P_0(z) - P_{N,0}z^N\left(f(z) + \left[\frac{-y}{x\lambda} - 1\right]\right)\right] \quad (31)$$

From (20), (31) we have

$$P_0(z)\left(z - \mu_1 f(z) - (1-\mu_1)f(z)^2\right) = -P_{N,0}z^N\left(xf(z)\left[f(z) + \left(\frac{-y}{x\lambda} - 1\right)\right] + z\right) + \lambda z(z-1)p_{0,0} \quad (32)$$

Let $\zeta = 1 - \lambda$ (33)

The zeros of the LHS of equation (32) are $z_{1,2} = \frac{y + 2\lambda x\zeta \pm \left(y^2 - 4x\lambda\zeta\right)^{1/2}}{-2x\lambda^2}$, at these values of z, the RHS of equation (32) should be equal to zero because of analyticity of $P_0(z)$ (being a polynomial in z of degree less than or equal to N) [14]

$$P_{N,0} = \frac{\lambda z_1(z_1-1)p_{0,0}}{z_1^N\left(x f(z_1)\left[f(z_1) - \frac{y}{\lambda x} - 1\right] + z_1\right)} \quad (34)$$

Using the fact that $P(1) = 1$, we solve for $P_{0,0}$ and substitute that value in $P(z)$

From (31) and (32)

$$P(1) = p_{0,0} + \sum_{j=0}^{1} P_j(1) = p_{0,0} + \sum_{j=0}^{1}\sum_{n=1}^{\infty} p_{n,j} = 1. \quad (35)$$

$$\sum_{j=0}^{1} P_j(1) = P_0(1)(2 - F_1) + \frac{y}{\lambda}P_{N,0} \quad (36)$$

$$P_0(1) = \lim_{z\to 1}\frac{-P_{N,0}z^N\left(xf(z)\left[f(z) - \frac{y}{\lambda x} - 1\right] + z\right) + \lambda z(z-1)p_{0,0}}{\left(z - \mu_1 f(z) - (1-\mu_1)f(z)^2\right)} \quad (37)$$

$$P_0(1) = \frac{-P_{N,0}\left((\lambda - 2y) + N\left(x + \frac{1}{\lambda}\right)\right) + \lambda p_{0,0}}{1-\rho} \quad (38)$$

$$\rho = \lambda\bar{c} \quad (39)$$

ρ is the load of the finite queueing system with $\bar{c}$ being the average decoding time.

$$\bar{c} = \sum_{j=0}^{2} jc_j = 2(1 - F_1) + \sum_{j=0}^{1} jc_j \quad (40)$$

From (5), (39), and (40): $\rho = \lambda(2 - F_1)$ (41)

Let $P_{N,0} = W_1 p_{0,0}$ (42)

where, $W_1 = \frac{\lambda z_1(z_1-1)}{z_1^N\left(x f(z_1)\left[f(z_1) - \frac{y}{\lambda x} - 1\right] + z_1\right)}$ (43)

Let $P_0(1) = W_2 P_{0,0}$ (44)

where $W_2 = \frac{-W_1\left((\lambda - 2y) + N\left(x + \frac{1}{\lambda}\right)\right) + \lambda}{1-\rho}$ (45)

From (35), (36), (41), (42), and (44) we have:

$$p_{0,0} = \frac{\lambda}{\left(\lambda + W_2\rho + yW_1\right)} \quad (46)$$

It is important as a performance wise to come up with derived expressions for the blocking probability (BP) and system throughput (ST). BP is the probability of packets of being dropped due to limited buffer size. (ST) is the average number of packets that get decoded per time slot.

$$BP = \sum_{j=0}^{1} P_{N,j} \quad (47)$$

From (12), (27), (42), (43), (44), (45), and (46) we have

$$\sum_{j=0}^{1} P_{N,j} = \frac{W_1\lambda}{\left(\lambda + W_2\rho + yW_1\right)}\left(1 - \frac{y}{\lambda}\right) \quad (48)$$

$$ST = \lambda\left(1 - \sum_{j=0}^{1} P_{N,j}\right) = \lambda - \frac{W_1\lambda(\lambda - y)}{\left(\lambda + W_2\rho + yW_1\right)} \quad (49)$$

We have $P(z) = p_{0,0} + \sum_{j=0}^{1} P_j(z).$

and the average number of packets in the buffer is the derivative of P(z) at z = 1 [1] [13]

From (31) and (32):

$$\sum_{j=0}^{1} P_j(z) = P_0(z)\left(1 + xf(z)\right) - xP_{N,0}z^N\left(f(z) - \frac{y}{\lambda x} - 1\right) \quad (50)$$

$$\frac{d}{dz}(P(z)) = x\lambda P_0(z) + (1 + xf(z))P_0'(z) - xP_{N,0}z^N(\lambda) - xNP_{N,0}z^{N-1}\left(f(z) - \frac{y}{\lambda x} - 1\right) \quad (51)$$

$$\left.\frac{d}{dz}(P(z))\right|_{z=1} = x\lambda P_0(1) + (1+x)P_0'(1) + P_{N,0}\left(N\left(\frac{y}{\lambda}\right) - x\lambda\right) \quad (52)$$

From (32) we have $P_0(z) = \frac{h_1(z) + \lambda z(z-1)p_{0,0}}{h_2(z)}$ (53)

$$P_0'(z) = \frac{h_2(z)\left(h_1'(z) + \lambda(2z-1)p_{0,0}\right) - \left(h_1(z) + \lambda z(z-1)p_{0,0}\right)h_2'(z)}{\left(h_2(z)\right)^2} \quad (54)$$

$$P_0'(1) = \lim_{z\to 1}\frac{h_2(z)\left(h_1'(z) + \lambda(2z-1)p_{0,0}\right) - \left(h_1(z) + \lambda z(z-1)p_{0,0}\right)h_2'(z)}{\left(h_2(z)\right)^2} \quad (55)$$

By taking the limit another time:

$$P_0'(1) = \lim_{z\to 1}\frac{h_2(z)\left(h_1''(z) + 2\lambda z p_{0,0}\right) - \left(h_1(z) + \lambda z(z-1)p_{0,0}\right)h_2''(z)}{2\left(h_2(z)\right)h_2'(z)} \quad (56)$$

$$P_0'(1) = \frac{h_2(1)\left(h_1'''(1) + 2\lambda p_{0,0}\right) + \left(h_1''(1) + 2\lambda p_{0,0}\right)h_2'(1) - W_2}{2\left(h_2(1)\right)h_2''(1) + 2\left(h_2'(1)\right)h_2'(1)} \quad (57)$$

where,

$$W_2 = \left(h_1(1)\right)h_2'''(1) + h_2''(1)\left(h_1'(1) + \lambda p_{0,0}\right) \quad (58)$$

$$h_2(z) = \left(z - \mu_1 f(z) - (1-\mu_1)f(z)^2\right) \quad (59)$$

$$h_2(1) = 0 \quad (60)$$

$$h_2'(1) = 1 - \rho \quad (61)$$

$$h_2''(1) = -2x\lambda^2 \quad (62)$$

$$h_2'''(1) = 0 \tag{63}$$

After substituting (58) - (63) in (57) we can get:

$$P_0'(1) = \frac{\left(h_1''(1) + 2\lambda p_{0,0}\right)(1-\rho) + 2x\lambda^2\left(h_1'(1) + \lambda p_{0,0}\right)}{2(1-\rho)^2} \tag{64}$$

$$h_1(z) = -P_{N,0} z^N \left(x f(z)\left[f(z) - \frac{y}{\lambda x} - 1 \right] + z \right) \tag{65}$$

$$h_1(1) = -P_{N,0}\left(1 - \frac{y}{\lambda}\right) \tag{66}$$

$$h_1'(1) = -P_{N,0}\left((\lambda - 2y) + N\left(x + \frac{1}{\lambda} \right) \right) \tag{67}$$

$$h_1''(1) = -P_{N,0}\left(2x\lambda^2 + 2N(\lambda - 2y) + N(N-1)\left(x + \frac{1}{\lambda} \right) \right) \tag{68}$$

After substituting (67), and (68) on (64) and then on (52), we can get the average buffer size.

It is interesting to find the average buffer occupancy when considering just one slot time for decoding. If decoding takes more than one slot, timeout occurs and the packet is discarded. Figure 4 shows the probability state transitions for this consideration. Basically we can get these probability state transitions from Figure 2 where $\mu_1 = 1$ (i.e. the conditional probability that decoding is completed in 1 slot given that the decoding is longer than 0 slots) and any unreachable state will be considered to zero like $p_{n,0}$, and $p_{n,1}$ where $n \geq 2$.

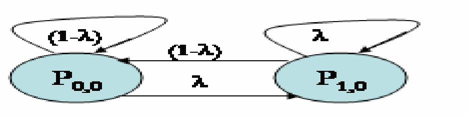

Figure 4: Probability state transitions for Geometric/Pareto/1/N queue with one slot time for decoding

The steady-state transition equations are:

$$P_{0,0} = 1 - \lambda\,,\ \ P_{1,0} = \lambda \tag{69}$$

$$\text{The average buffer occupancy} = \sum_{n=0}^{1} n p_{n,0} = \lambda \tag{70}$$

The average buffer occupancy when considering just one slot for decoding increase linearly with λ and it is independent of the Maximum buffer capacity (since at least the buffer capacity is assumed to be one) and Pareto exponent. This is to be expected since each packet can only spend one slot for decoding or time-out occurs.

C. Analytical Results

Figure 5 shows the blocking probability versus buffer capacity under different channel conditions and packet arriving probabilities. It is obvious from Figure 5 for a fixed packet arriving probability (λ) the blocking probability increases as β decreases. It is also shown that at a fixed channel condition (β) the blocking probability increases as λ increases.

Figure 6 shows the system throughput that is the average number of data packets that are decoded per time slot versus buffer capacity under different channel conditions and packet arriving probabilities. It is shown in Figure 6 that the system throughput can reach packet arriving probability (λ) as buffer capacity increases more and more.

It is interesting to see the average buffer occupancy from Figure 7 when considering just one slot for decoding (DS = 1). The linear relationship with a slope λ and intercept zero is noticed (Note that the log scale is used for the buffer occupancy in Figure 7). This linear relationship is also independent of the Pareto exponent, $\beta = x$ (don't care), and buffer capacity, N = x (don't care), this is to be expected since each packet can only spend one slot for decoding or time-out occurs.

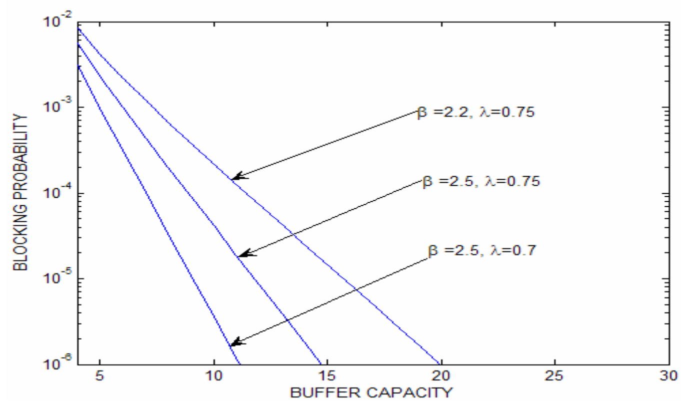

Figure 5. Blocking probability versus buffer capacity.

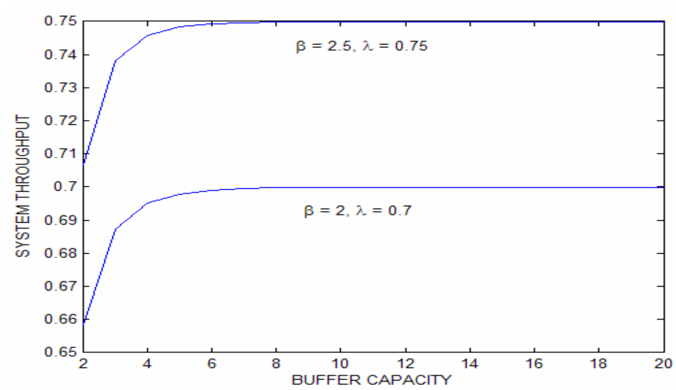

Figure 6. System throughput versus buffer capacity.

As a result can be also seen in Figure 7, given a fixed packet arrival rate and buffer capacity (100), the average buffer occupancy increases as β decreases. This is expected since a lower β means the channel is much noisy (more decoding time is required). The increase in the average buffer occupancy as the decoding slots (DS) increase can also be observed in Figure 7. A larger time-out limit may cause buffer overflow at a fixed arrival rate of incoming packets.

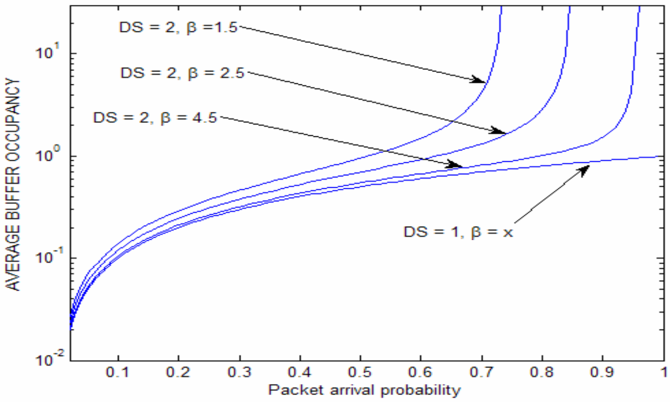

Figure 7. Average buffer occupancy versus packet arrival probability λ

IV. SIMULATION

The simulation for Geometric/Pareto/1/N model is done using MATLAB. The purpose of this simulation in this paper is to measure just the average buffer occupancy performance metric. In our previous work [11] the simulation for average buffer size and some other performance metrics evaluations for different decoding slots has been performed with some different assumptions, but for this paper new simulation has been done with the same assumptions as our analytical analysis. The new simulation has been performed just for decoding slots (one or two) and for different channel conditions and buffer capacities to be then compared with our analytical results. The sequential decoding system is simulated for wireless multimedia networking systems that work on connectionless mode and present best effort service where there is no handshaking between sender and receiver. For real-time video streaming, UDP is usually preferred for video streaming where resending lost packets is not practical [21]. The deadline to display packets cannot be met by resending the packets.

In our analysis, we assumed that each slot is equal to the packet transmission time. We assume that all the incoming packets have equal length (as in ATM networks or wireless links). Consequently, the decoder can receive at most one new packet during a slot. However, a random number generator for Bernoulli distribution is invoked at the beginning of every time slot to demonstrate the arrival packets. A random number generator for Pareto distribution is invoked at the beginning of any time slot as long as there are packets in the queue waiting for service to demonstrate the heavy tailed service times. The minimum service time is assumed to be one. The decoding slots (timeout limit) and buffer capacity (length or size) are taken as inputs of the simulation.

Service or decoding can begin only at the start of the next time slot of arriving packet if there are no packets waiting in the queue, i.e. every arrival packet has to stay in the queue (if there is an empty room) for at least one time slot. A time history vector with length equal to simulation time is updated at every time slot. Every entry in that time history vector means the number of packets in the buffer waiting for service at that time slot (i.e. state of the queue). A state of the queue is managed and updated at every time slot until to the end of the simulation time since every slot time there might be an arrival packet (this packet may get lost due to buffer overflow) and one packet gets the service at the same time. The average buffer occupancy is the average or mean of this time history vector.

A. *Simulating the Distributions*

The Bernoulli distribution is a discrete distribution having two possible outcomes labeled by t (= 0 or 1). t = 1 (packet arrival) occurs with probability λ, and t = 0 (no packet arrival) occurs with probability $1-\lambda$, where $0 \le \lambda \le 1$. Thus, the probability mass function of the Bernoulli distribution is: $P(t) = \begin{cases} 1-\lambda & \text{for } t = 0, \\ \lambda & \text{for } t = 1. \end{cases}$ The Bernoulli random number n can be generated by generating the well-known u ~ unif (0,1), where u is uniformly distributed in the region of [0,1], and applying "If u < $1-\lambda$ then t = 0"; Otherwise "t = 1". Furthermore, the Pareto distribution is a heavy-tailed distribution with parameters k and β. Since there is no random number generator available for the Pareto distribution in Matlab, we use the Inverse CDF (cumulative distribution function) method [3] [11] to generate a Pareto distributed random number by generating u ~ unif (0,1) just like before and then computing. $y = F_\tau^{-1}(u)$, where $F_\tau^{-1}(\tau)$ is the inverse CDF of the Pareto distribution.

It is well known that the CDF of the Pareto distribution is $F_\tau(\tau) = 1 - \left(\frac{k}{\tau}\right)^\beta$, where k is the minimum decoding time, and β is a function of the SNR of the channel [3]. Therefore, its inverse function can be found as

$$F_\tau^{-1}(\tau) = \frac{k}{(1-\tau)^{\frac{1}{\beta}}} . \tag{71}$$

Now it is easy to generate a Pareto random number y by using

$$y = F_\tau^{-1}(u) = \frac{k}{(1-u)^{\frac{1}{\beta}}} . \tag{72}$$

B. *Simulation Results*

The consistency between Figures 7 and 8 are noticed. Whenever the simulation time is increased more and more, closer and smooth results are obtained.