Queuing Analysis and Simulation of Wireless Access and End Point Systems using Fano Decoding

Khalid A. Darabkh
The University of Jordan/Computer Engineering Department, Amman, Jordan
Email: k.darabkeh@ju.edu.jo

Abstract— **Wireless networks are a growing technology due to its ability to receive data in areas where it is very hard to plug-in using wires. TCP Reno assumes in his congestion algorithms that the packet loss is the major cause of network congestion. In wireless networks, this is not correct because having a high bit error rate leads also to a packet loss. Link layer approach is one of the most efficient proposed solutions to maintain TCP over wireless networks. For example, having hybrid ARQ type 1 with Fano decoding, which is an error correction technique, is very appropriate and is of concern in wireless networks due to its capability of offering decoding steps, which are dependent to the channel state. In this paper, we propose a novel queuing model to see the effect of employing Fano decoding on the buffer of wireless access or end points since it is a very effective network parameter and cannot be neglected. Our queuing model is concerned not only about those departed packets after being decoded using Fano algorithm, but also the way packets arrive to the wireless access or end point systems. An analytical study has been conducted to derive a general form expression for the average number of packets residing in the system's buffer. On the other hand, a simulation study using programming language has been performed to validate our analytical results.**

Index Terms— **Expected queue size, Fano decoding, partial decoding, queuing analysis and simulation, TCP performance.**

I. Introduction

Wireless networks have become very popular nowadays for a lot of reasons, most importantly the flexibility to set up a network where it is difficult using wired networks. TCP (transmission control protocol) is the main protocol to deliver the service with completeness, no errors, and fast delivery enhancements. It was originally designed for wired networks. TCP Reno is the typical congestion control mechanism that is implemented by end hosts. TCP congestion schemes limit the connection throughput by interpreting the loss of the packet as an indication of congestion [1], [2]. This is correct for wired networks where the transmission media is noiseless, but in wireless networks the media is noisy by nature [3]. Even low BER (bit-error-rate) may generate packet loss as a result of DUP ACKs (duplicate acknowledgments) and timeout. Consequently, the sender, which is not wireless-aware, shrinks its congestion window due to a congestion mark understanding. Hence, multiple corrupted packets may decrease the sending rate of the TCP sender dramatically.

Manuscript received January 31, 2010; revised April 18, 2010; accepted May 11, 2010.

The following are the major approaches that have been done to improve TCP over wireless networks [4]: Split mode, Snoop protocol, and link layer. In the split mode, there are two separate TCP streams. In this approach, a certain packet may be acknowledged by the access point without reaching its destination. This violates the semantic of TCP as being an end-to-end scheme [2]. This further leads to extra overhead for wireless access points by adding a new layer (i.e., transport layer) to manage that. In snooping protocol, the idea of ELN (explicit loss notification) has been introduced [5]. This is done by activating an ELN flag in the TCP header to distinguish DUP ACKs received by sender. It is set when these DUP ACKs come as a result of a noisy environment. In this case, the congestion algorithms will not be invoked and the congestion window will be kept large. It works efficiently only when the error rate is low [6]. Moreover, it has a limitation of possibly false notification, especially when there are multiple packets lost in a window [6]. In the last approach, a set of algorithms have been associated with ARQ (automatic repeat request) (i.e., hybrid ARQ of type 1) to enhance the TCP performance over wireless networks through the MAC (media access control) sub-layer of wireless access and end points and most importantly the convolutional decoding algorithms [4]. These decoding algorithms are considered error detection and correction algorithms and are of interest in wireless networks much more than the CRC (cyclic redundancy check) error detection technique. In this paper, we investigate the last approach from a queuing point of view. In other words, we study the effect of employing decoding algorithms on the buffer of access or end points since it is a very important network parameter and may affect the network performance badly if it is not taken into consideration.

Convolutional codes are a category of channel coding that are described by adding extra bits to the original data for bit flipping prevention over a noisy channel [7]. A convolutional coder includes L-stage shift register and x codeword blocks modulo-2 adders [8]. Hence, it has a constraint length of L. There are two important decoding (error detection and correction) algorithms for convolutional codes, the maximum-likehood decoding

(Viterbi's algorithm) and sequential decoding [8]. These algorithms increase the ratio of good packets to the total number of received packets. Moreover, they may decrease overall message latency since correcting corrupted packets helps to reduce the number of retransmissions and accordingly network congestion. Consequently, the TCP performance over wireless networks is improved. Convolutional coding with Viterbi decoding is very popular and is considered the predominant FEC (forward error correction) technique that is widely employed in satellite and mobile communications [8], [9]. It is characterized by affording a fixed decoding time. Therefore, it is unable to provide faster decoding for a received codeword sequence that contains fewer errors.

Fano decoding was developed by Fano [9] and came as an efficiency improvement to a sequential decoding algorithm that was developed by Wozencraft [9]. Fano algorithm has advantages over Viterbi decoding algorithm because it is operated at a variable decoding rate and has computational and storage requirements that grow linearly as a function of the constraint length rather than exponentially as the case of Viterbi decoding [10]. Moreover, it has been proven that under high SNR (signal-to-noise ratio), the Fano decoding consumes less power than the Viterbi algorithm [11]. Actually, the complexity of Fano algorithm becomes dependent on the channel state (noisy or pure) [10], [12], and [13]. The brief description of this algorithm is as follows [10]: Once the codeword sequence is received by the decoder, a comparison is made with the codeword allowed by the decoder according to the encoder state diagram. Each codeword sequence is divided into groups where each group consists of *m* digits. Fano algorithm works in a way similar to the code tree. It chooses the path for which the sequence group has the shortest hamming distance (i.e., difference in bits between encoder output and the received sequence group is minimum). This process is repeated until groups end. If a lot of errors appear through this process then it becomes impossible to have a match for the received sequence. This is an indication that the wrong path is chosen. Thus, back and forth steps have to be done to avoid getting an accumulatively high number of errors.

It is well known that every intermediate hop has a buffer to absorb the variable packet processing rate. In this paper, we propose a new queuing model that describes packet arrivals and departures of a wireless system uses Fano decoding. The maximum decoding time has to be variable since it depends on how high the percentage of errors is. In our queuing model, the decoder is described as having not only an upper variable decoding limit (*T*), but also a lower variable decoding limit (*K*). The minimum decoding time of Fano decoding is also variable (i.e., not constant) since it depends on how low the percentage of errors is. We consider in our queuing model that the channel state is also variable. Having a Fano decoder with upper and lower operating decoding limits that are variable and adaptive with the channel state is very efficient and suitable for noisy media such as wireless networks. Furthermore, it gives generality for our system in terms of queuing theory. Our major aim of this queuing model is to find a closed form expression of the average number of packets waiting in the system's buffer. The system is assumed to be of infinite capacity. Hence, it becomes very necessary to find the average number of packets in the buffer waiting to be served so that it can be selected as a value to be operated on. It is clear that the value of system capacity (buffer size) cannot be chosen randomly since choosing too large or too small may degrade the performance of the system significantly. The average packet waiting time increases when the system capacity gets large. Thus, many retransmissions may occur due to a timeout limit from the sender side although there is no packet loss. On the other hand, many retransmissions may occur due to packets being discarded because of small system capacity. Hence, choosing a suitable buffer size may also improve TCP performance over wireless networks. Our queuing model works with not only Fano decoding, but also any other decoding algorithm of variable complexity such as Low-density parity-check and Turbo decoders. We also provide an expression of the maximum possible packet arriving probability in order to keep the decoder's buffer stable, which then is verified through the results of our general form expression of the average buffer size. Moreover, we simulate our proposed queuing model using Matlab programming to validate our analytical observations and results. To the best of our knowledge, our queuing model, analysis, simulation, and results are completely new and there is no similar work to ours.

II. ANAYTICAL STUDY

In our analytical study, we provides details about the system model, system assumptions and probability state transitions, system steady-state transition probabilities, and system analysis that are very helpful to recognize our proposed general form expression for the average queue size.

A. System Model

To model the system (decoder and queue), a discrete-time Markov chain is used. A Markov chain is a stochastic process [14] (bunch of random variables) with a very limited memory [14]. However, in our model, the time is portioned into equal time slots where at least a packet is allowed to arrive in any arbitrary time slot. Hence, Bernoulli process is the best to describe arriving packets. The arriving probability arrives with probability (λ) and does not with probability $(1-\lambda)$. The decoder can start decoding a packet after its arrival immediately on the succeeding slot if there are no packets waiting in the queue. Since Fano algorithm offers variable decoding time which is dependent to the channel state, Pareto distribution, which is a heavy-tailed distribution, is considered to be the best to describe the decoding time [15]-[17]. In this distribution, there is a parameter called (β). When the value of this parameter gets high, the PDF (probability density function) of Pareto distribution goes to zero early. From the decoder side, this means dispatching low decoding time. On the other hand, when (β) gets low, the PDF goes to zero late. Considering decoder side, this implies dispatching high decoding time. Hence, we can take advantage of that by connecting (β) to the SNR of the channel [12], [18]. Thus, when SNR is high, this indicates we have low noise power. Therefore, low decoding time is required. We will lose the system's generality if this lower decoding time is considered to be fixed as assumed in [19]. However, when SNR is low, this implies that the noise power is high. Thus, high decoding time is needed. However, once a packet reaches that upper bound decoding limit and needs more, it is considered to be partially decoded and lost which is the same as assumption used in [19]-[21]. One way of recovering those packets is to resend them during the slots follow immediately getting partial decoding [20], [22].

B. System Assumptions and Probability State Transitions

The states of this discrete-time Markov model are represented by (*n, j*) as employed in [19], [23], where *n* describes the number of packets in the system including the packet being decoded, and *j* stands for the number of slots elapsed in decoding the packet currently in service. We assume two major notations to understand the probability state transitions shown in Fig. 1. The probability that the decoding process is completed in *j+1* slots is referred to c_{j+1} and the probability that the decoder finishes decoding in *j+1* slots given that decoding takes more than *j* slots is notated as μ_j. Thus, the conditional probability is:

$$\mu_j = \frac{c_{j+1}}{(1-F_j)}. \tag{1}$$

Where F_j refers to the distribution of decoding time and it can be represented as follows:

$$F_j = 1 - P_F(jT_r), \quad 1 \le j \le T, \tag{2}$$

Where T_r is the slot duration and assumed to be 1, *T* is the maximum decoding slots allowed for decoding. Hence,

$$P_F(j) = \Pr\{t > j\} = \left(\frac{j}{2}\right)^{-\beta}. \tag{3}$$

Note that the decoding time of Fano decoders has the Pareto distribution where β is called the Pareto parameter and it is a function of SNR [1], [20] of the wireless channel which means that if SNR gets low, then the wireless channel gets worse. However, 2 represents the minimum decoding time needed by the decoder to decode a packet.

$$F_j = \sum_{i=2}^{j} c_i, \quad j \ge 2. \tag{4}$$

To describe the behavior of Fano decoding using queuing theory with high level of generality, the Pareto parameter, packet arriving probability, maximum decoding limit, and minimum decoding limit should be variable. We have already introduced elementary analytical results for the average buffer size but for the special case where at least four decoding time slots are required to finish decoding [24]. In this work, we start first by assuming that packets need at least two decoding time slots to finish decoding (i.e., minimum decoding limit) as shown in (1) and (3). Consequently, we become able to generalize our close form expression for the average buffer size to be a function of upper variable decoding limit (*T*), channel condition (β), packet arriving probability (λ) in addition to lower variable decoding limit (*K*), which is presented in subsection *E*. We present in this work to the extent level details of how to reach that general form. It is important to state that the number of state probabilities increases when the upper variable decoding limit (*T*) increases (as shown in Fig. 1) while it is not the case for lower variable decoding limit since it has just a relation with μ_j. Therefore, the step transitions always get different when that lower limit is changed.

According to our initial assumption saying that any packet needs at least two decoding slots,

$$F_0 = 0. \tag{5}$$

$$F_1 = 0. \tag{6}$$

It can clearly be shown that,

$$1 - F_j = \prod_{i=2}^{j} (1-\mu_{i-1}), \quad j \ge 2. \tag{7}$$

And the probability c_j is given by

$$c_j = \begin{cases} F_j - F_{j-1}, & 2 \le j \le T, \\ 0, & \textit{Otherwise.} \end{cases} \tag{8}$$

C. System Steady-state Transition Equations

All states found in Fig. 1 are neither transient nor null recurrent. Thus, stationary (limiting, equilibrium, or steady-state) probabilities exist [14]. Let $p_{n,j}$ is the probability that the decoder has *n* packets in its queue including the packet being decoded which is in the *j*th slot of decoding. Accordingly, the steady-state transition equations are given below.

$$p_{0,0} = (1-\lambda)p_{0,0} + \sum_{j=1}^{T-1} \mu_j (1-\lambda) p_{1,j}, \tag{9}$$

$$p_{0,j} = 0, \tag{10}$$

$$p_{1,0} = \sum_{j=1}^{T-1} \mu_j \left[\lambda p_{1,j} + (1-\lambda) p_{2,j}\right] + \lambda p_{0,0}, \tag{11}$$

$$p_{1,1} = (1-\lambda) p_{1,0}, \tag{12}$$

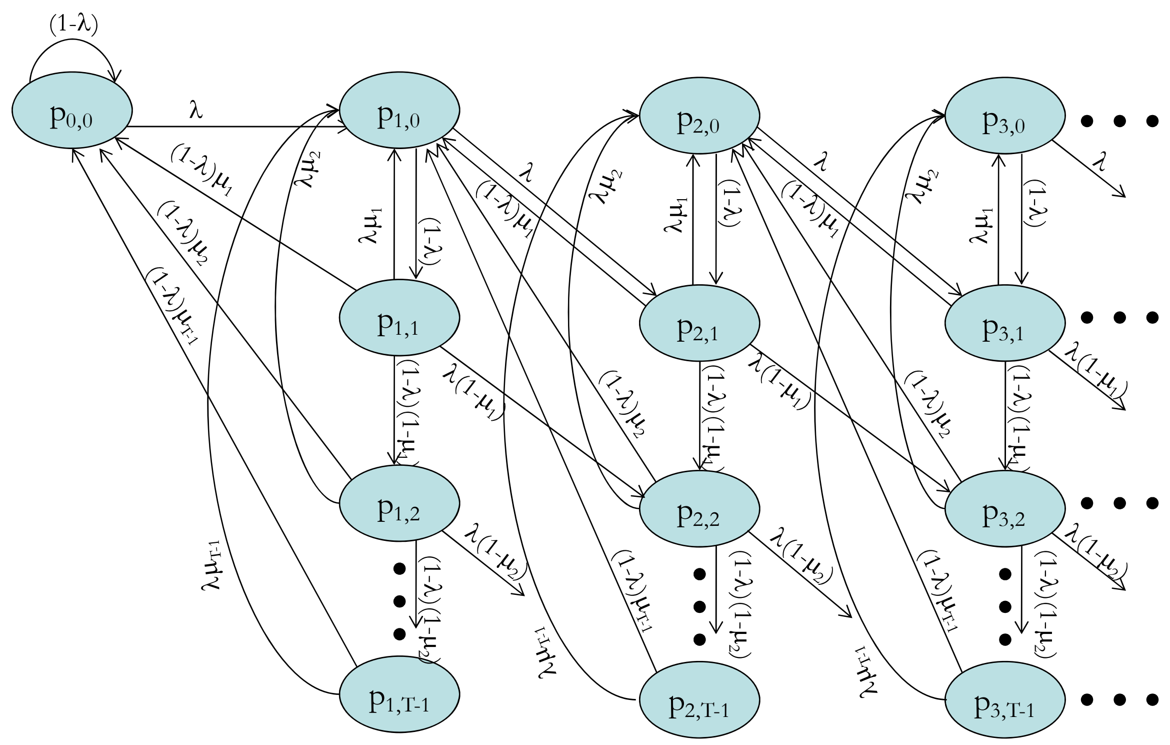

Figure 1: Probability state transitions of decoder's queue with maximum and minimum decoding limits of T and 2 respectively. The summation of outgoing links of all states is equal to one.

$$p_{1,j} = (1-\lambda)(1-\mu_{j-1})p_{1,j-1}, \qquad 2 \le j \le T-1, \tag{13}$$

$$p_{n,0} = \sum_{j=1}^{T-1} \mu_j \left[\lambda p_{n,j} + (1-\lambda)p_{n+1,j}\right], \qquad 2 \le n, \tag{14}$$

$$p_{n,1} = \lambda p_{n-1,0} + (1-\lambda)p_{n,0}, \qquad 2 \le n, \tag{15}$$

$$p_{n,j} = (1-\mu_{j-1})\left[(1-\lambda)p_{n,j-1} + \lambda p_{n-1,j-1}\right],$$
$$2 \le n, \quad 2 \le j \le T-1, \tag{16}$$

Taking into consideration that the conditional probability to finish decoding in T slots given that at least required $T-1$ slots, which refers to μ_{T-1}, is equal to one.

D. System Analysis

According to Fig. 1, dependency between states is visible. To solve the problem of discovering intelligent way to find the average number of packet in the queue, the following generating functions are defined:

$$P(z) = p_{0,0} + P_0(z) + P_1(z) + \sum_{j=2}^{T-1} P_j(z), \quad 2 \le j \le T-1. \tag{17}$$

Where,

$$P_i(z) = \sum_{n=1}^{\infty} p_{n,i} z^n, \quad 0 \le i \le T-1. \tag{18}$$

and $P(z)$ is the generating function that denotes for the number of packets in the queue waiting to get served. It is important to refer to the well known fact that $P(1) = 1$ [20], [25]. Thus, we can evaluate the average number of packets in the queue by taking the derivative of $P(z)$ at $z = 1$ [19], [20], [25], and [26]. However, referring to (18), we can write

$$P_0(z) = p_{1,0} z + \sum_{n=2}^{\infty} p_{n,0} z^n. \tag{19}$$

Substituting (11) and (14) for $p_{1,0}$ and $p_{n,0}$ into (19), we get

$$P_0(z) = \sum_{n=1}^{\infty}\sum_{j=1}^{T-1} \mu_j \left[\lambda p_{n,j} + (1-\lambda)p_{n+1,j}\right] z^n + \lambda z p_{0,0}. \tag{20}$$

After rewriting, we get

$$P_0(z) = \sum_{j=1}^{T-1} \mu_j \left[\lambda \sum_{n=1}^{\infty} p_{n,j} + (1-\lambda)\sum_{n=1}^{\infty} p_{n+1,j}\right] z^n + \lambda z p_{0,0}. \tag{21}$$

The rightmost summation can be written as

$$\sum_{n=1}^{\infty} p_{n+1,j} z^n = \frac{1}{z}\left(P_j(z) - p_{1,j} z\right). \tag{22}$$

Assuming,

$$f(z) = 1 - \lambda + z\lambda. \tag{23}$$

After finding the term $\sum_{j=1}^{T-1} \mu_j (1-\lambda) p_{1,j}$, found at (9), in terms of $p_{0,0}$ and substituting it along with (22) and (23) into (21), we find

$$P_0(z) = \frac{f(z)}{z} \sum_{j=1}^{T-1} \mu_j P_j(z) + \lambda(z-1) p_{0,0}. \tag{24}$$

Applying the fact that $\mu_{T-1} = 1$,

$$P_0(z) = \frac{f(z)}{z}\left[P_{T-1}(z) + \sum_{j=1}^{T-2}\mu_j P_j(z)\right] + \lambda(z-1)p_{0,0}. \quad (25)$$

For $P_j(z)$ where $j=1$ we have,

$$P_1(z) = p_{1,1}z + \sum_{n=2}^{\infty} p_{n,1}z^n. \quad (26)$$

Substituting (12) and (15) into (26) yields

$$P_1(z) = (1-\lambda)p_{1,0}z + \sum_{n=2}^{\infty}\left[\lambda p_{n-1,0} + (1-\lambda)p_{n,0}\right]z^n. \quad (27)$$

We can rewrite (27) as follows

$$P_1(z) = \lambda\sum_{n=2}^{\infty} p_{n-1,0}z^n + (1-\lambda)\sum_{n=1}^{\infty} p_{n,0}z^n. \quad (28)$$

The leftmost summation can be simplified as

$$\sum_{n=2}^{\infty} p_{n-1,0}z^n = z\sum_{n=2}^{\infty} p_{n-1,0}z^{n-1} = zP_0(z). \quad (29)$$

Hence,

$$P_1(z) = f(z)P_0(z). \quad (30)$$

Deriving an expression for $j>1$,

$$P_j(z) = p_{1,j}z + \sum_{n=2}^{\infty} p_{n,j}z^n. \quad (31)$$

Substituting (13) and (16) into (31) gives

$$\begin{aligned} P_j(z) &= (1-\lambda)(1-\mu_{j-1})p_{1,j-1}z \\ &+ (1-\mu_{j-1})\sum_{n=2}^{\infty}\left[(1-\lambda)p_{n,j-1} + \lambda p_{n-1,j-1}\right]z^n. \end{aligned} \quad (32)$$

After rearranging, we get

$$\begin{aligned} P_j(z) &= (1-\lambda)(1-\mu_{j-1})\sum_{n=1}^{\infty} p_{n,j-1}z^n \\ &+ \lambda(1-\mu_{j-1})\sum_{n=2}^{\infty} p_{n-1,j-1}z^n. \end{aligned} \quad (33)$$

Utilizing the same principle used in (22) and (29), we get

$$P_j(z) = (1-\mu_{j-1})f(z)P_{j-1}(z). \quad (34)$$

Applying (7), we recursively get

$$P_j(z) = (1-F_j)f(z)^j P_0(z). \quad (35)$$

Substituting (35) into (24) and applying (1) yields

$$P_0(z) = \frac{q(z)}{z}P_0(z) + \lambda(z-1)p_{0,0}. \quad (36)$$

Where,

$$q(z) = \sum_{j=1}^{T-2} c_{j+1}f(z)^{j+1} + (1-F_{T-1})f(z)^T. \quad (37)$$

Finally, we get

$$P_0(z) = \frac{\lambda z(z-1)}{z-q(z)}p_{0,0} = y(z)p_{0,0}. \quad (38)$$

After substituting (30), (35), and (38) into (17), we get

$$P(z) = \left[1 + y(z) + f(z)y(z) + \sum_{j=2}^{T-1}(1-F_j)f(z)^j y(z)\right]p_{0,0}. \quad (39)$$

We can get the value of $p_{0,0}$ after applying the fact that $P(1)=1$ on (17). Hence, we get

$$p_{0,0} = \frac{1}{\left[1 + y(1) + f(1)y(1) + \sum_{j=2}^{T-1}(1-F_j)f(1)^j y(1)\right]}, \quad (40)$$

Where,

$$y(1) = \lim_{z\to 1}\frac{\lambda z(z-1)}{z-q(z)}, \quad (41)$$

Substituting $z=1$ into (23) and (37), and considering (4), we get

$$q(1) = 1, \quad (42)$$

Hence,

$$y(1) = \lim_{z\to 1}\frac{(2z-1)\lambda}{1-\lambda\left(\sum_{j=1}^{T-2}(j+1)c_{j+1}f(z)^j + T(1-F_{T-1})f(z)^{T-1}\right)}, \quad (43)$$

After applying L'Hospital's law just once, we conclude

$$y(1) = \frac{\lambda}{1-\lambda\tau}, \quad (44)$$

Where,

$$\tau = \sum_{j=1}^{T-2}(j+1)c_{j+1} + T(1-F_{T-1}), \quad (45)$$

Thus, after substituting (44) into (40), we finally conclude

$$p_{0,0} = \frac{1}{\left[1 + \frac{\lambda(2+\eta)}{1-\lambda\tau}\right]}, \quad (46)$$

Where,

$$\eta = \sum_{j=2}^{T-1}(1-F_j). \quad (47)$$

The average number of packets in the decoder's queue is found by taking the derivative of (39) and then substituting for $z=1$. Hence, we get

$$P'(1) = \begin{bmatrix} y'(1) + f'(1)y(1) + f(1)y'(1) \\ + \sum_{j=2}^{T-1}(1-F_j)\left[jf(1)^{j-1}f'(1)y(1) + f(1)^j y'(1)\right] \end{bmatrix} p_{0,0}, \quad (48)$$

After taking the derivative of $y(z)$ found at (38), we get

$$y'(1) = \lim_{z\to 1}\lambda\frac{[z-q(z)](2z-1) - [1-q'(z)]z(z-1)}{[z-q(z)]^2}, \quad (49)$$

By using L'Hospital's law twice, we get

$$y'(1) = \lambda\left(\frac{q''(1) + 2[1-q'(1)]}{2[1-q'(1)]^2}\right), \quad (50)$$

Taking the derivative of (37) and then substituting for $z=1$ when using (23)

$$q'(1) = \lambda\left(\sum_{j=1}^{T-2}c_{j+1}(j+1) + T(1-F_{T-1})\right), \quad (51)$$

Also for the second derivative when applying further the fact that $f''(z)=0$

$$q''(1)=\lambda^2\left(\sum_{j=1}^{T-2}(j+1)jc_{j+1}+T(T-1)(1-F_{T-1})\right), \quad (52)$$

Hence, we finally get

$$y'(1)=\lambda\left(\frac{\lambda^2\bar{\tau}+2[1-\lambda\tau]}{2[1-\lambda\tau]^2}\right), \quad (53)$$

Where,

$$\bar{\tau}=\sum_{j=1}^{T-2}j(j+1)c_{j+1}+T(T-1)(1-F_{T-1}). \quad (54)$$

Substituting (44), (46), and (53) into (48) leads to the final expression of the average number of packets in the buffer

$$P'(1)=\frac{(1-\lambda\tau)\left[y'(1)(2+\eta)+\lambda y(1)\left(1+\bar{\eta}\right)\right]}{1+\lambda(2+\eta-\tau)}, \quad (55)$$

Where,

$$\bar{\eta}=\sum_{j=2}^{T-1}j(1-F_j). \quad (56)$$

E. Average Buffer Size with Variable Lower Bound Decoding Limit

Due to the nature of wireless links of being uncontrollable and noisy (i.e., high BER) because of many different conditions such as multipath interference, urban obstacles, mobility of wireless end points, large moving objects, as well as weather conditions, it becomes interesting to derive further a new general form of average number of packets in the buffer of a decoder that is also bounded by variable K which denotes to minimum decoding limit. With refereeing to our analysis in subsection D (where K=2) and our previous analysis [24] (for K=4), we conclude that (39) will be modified to include any value of K as follows,

$$P(z)=\begin{bmatrix}1+y(z)+f(z)y(z)+...+f(z)^{K-1}y(z)\\ +\sum_{j=K}^{T-1}(1-F_j)f(z)^j y(z)\end{bmatrix}p_{0,0}. \quad (57)$$

The term $(1-F_j)$ will be just associated with $P_j(z)$ when $j\geq K$ (for details, see how to derive (35)). When applying the fact that $P(1)=1$, we can get after updating (40) according to our previous modified equation (i.e., (57)) the following,

$$p_{0,0}=\frac{1}{\left[1+\frac{\lambda(K+\eta)}{1-\lambda\tau}\right]}, \quad (58)$$

Where,

$$\eta=\sum_{j=k}^{T-1}(1-F_j), \quad (59)$$

Finding $P'(1)$ from (57), we get the general expression for the average buffer size with complete variable parameters,

$$P'(1)=\left[y'(1)(K+\eta)+\lambda y(1)\left(K(K-1)-\sum_{i=1}^{K-1}i+\bar{\eta}\right)\right]p_{0,0}, \quad (60)$$

Where,

$$\tau=\sum_{j=1}^{T-K}(j+K-1)c_{j+K-1}+T(1-F_{T-1}), \quad (61)$$

$$y'(1)=\lambda\left(\frac{\lambda^2\bar{\tau}+2[1-\lambda\tau]}{2[1-\lambda\tau]^2}\right), \quad (62)$$

$$\bar{\tau}=\sum_{j=1}^{T-K}(j+K-1)(j+K-2)c_{j+K-1} + T(T-1)(1-F_{T-1}), \quad (63)$$

$$\bar{\eta}=\sum_{j=K}^{T-1}j(1-F_j), \quad (64)$$

And,

$$y(1)=\frac{\lambda}{1-\lambda\tau}. \quad (65)$$

The distribution of decoding time becomes as,

$$F_j=\begin{cases}\sum_{i=K}^{j}c_i, & j\geq K,\\ 0 & Otherwise.\end{cases} \quad (66)$$

Moreover,

$$1-F_j=\prod_{i=K}^{j}(1-\mu_{i-(K-1)}), \qquad j\geq K. \quad (67)$$

III. ANAYTICAL RESULTS

Fig. 2 shows the effect of operating different upper bound decoding limits, packet arriving probabilities, as well as channel conditions on the buffer size when working on a value of two for lower bound decoding limit (notice the *semilogy* shape adapted). However, the increase in the average number of packets, when packet arriving probability increases, is noticed. This can be explained by more arriving packets than the buffer can serve. The upper bound decoding limit has a noticed impact on the average number of packets in the buffer. Once this limit becomes larger, the number of packets waiting for decoding gets also larger. For example, the average buffer size, when $\lambda=0.0489$, is 528.3, 5.43, and 1.324 for $T=90, 60,$ and 30 respectively (given $\beta=0.6$). When comparing plots (a) with (b), the effect of changing the channel condition (β) is shown. The decrease in β means the channel gets worse. Thus, higher decoding time is needed. Consequently, all packets may reach the upper bound decoding limits and this leads to increase in the average number of packets waiting in the buffer. As values selected from Fig. 2 to verify that, the average buffer size, when $\lambda=0.2084$, is 333 and 5.622 for $\beta=0.2$ and 0.6 respectively (given $T=5$).

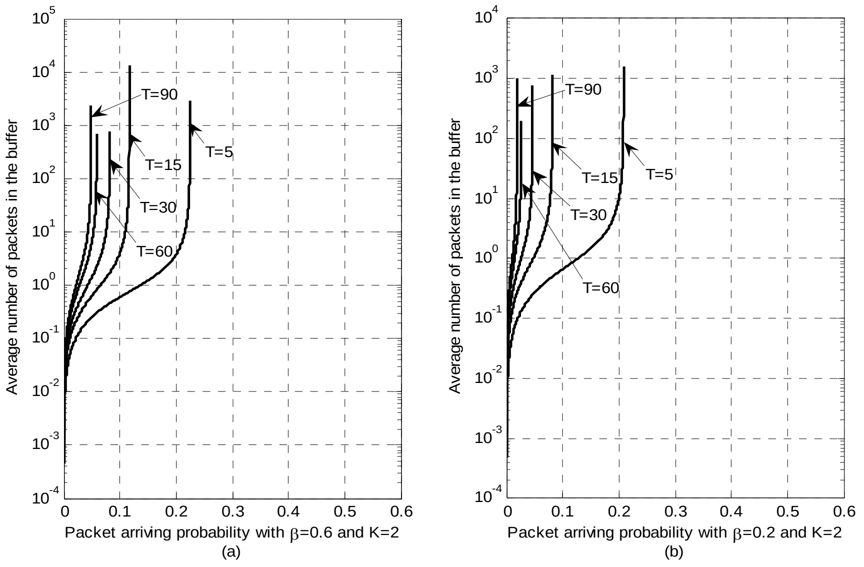

Figure 2: Average number of packets in the buffer versus packet arriving probability (λ) for different T and β. (a) $\beta = 0.6$. (b) $\beta = 0.2$.

Figs. 3 and 4 show the average number of packets in the decoder's buffer for different lower bound decoding limit (K), upper bound decoding limit (T), channel condition (β), and packet arriving probability (λ). Our major goal is to see the effect of changing K on the average buffer size for different values of $T, \lambda,$ and β. The increase in the average buffer size is clearly seen as long as K increases for fixed values of $T, \lambda,$ and β. This is expected since increasing K leads to having more packets in the buffer waiting for service. The decrease in β as seen in Fig. 3 causes a noticed increase in the average buffer size for all values of K. Table I provides values for the average buffer size when choosing different values of λ to be compared in both subplots of Fig. 3 under different K. As a summary, the average buffer size increases for any value of K when β decreases. In Fig. 4, the effect of decreasing T which leads to a decrease in the average buffer size is noticed when compared to Fig. 3 for the same values of $K, \lambda,$ and β. Table I also presents values for the average buffer size for different values of T (60 and 15). For fixed values of λ and β, the average buffer size decreases as T decreases for all values of K.

There is one more observation is shown in Table I (also can be extracted from Figs. 3 and 4). The value of average buffer size is 365.3 when $(T = 60, \beta = 1.7)$ and 6.304 when $(T = 15, \beta = 1.7)$ while the value of average buffer size is 1.744 when $(T = 60, \beta = 2.7)$ and 1.591 when $(T = 15, \beta = 2.7)$. Does that mean, for $\lambda = 0.19330$ and $K = 2$, we should employ a buffer size near 7 for $\beta = 1.7$ and 2 for $\beta = 2.7$ regardless of the value of T? The answer is yes for the case of $\beta = 2.7$ but is no for $\beta = 1.7$. It is yes because there no much difference obtained in the average buffer size and this is a good indication that the channel is not so noisy because of relatively high value of β. For the other part of the answer, which is no, and that is because if we select 7 when T=60, then many packets will get partially decoded and accordingly will be considered to be lost and need to be retransmitted since the buffer size is too small. This is an indication that the channel becomes so noisy since β is low. Thus, we can summarize that the value of T becomes less sensitive to the change of average buffer size when β gets larger.

It is shown from Figs. 2, 3, and 4 that the average number of packets goes to infinity after reaching certain limit of packet arriving probability (λ_{max}). The increment chosen for λ in Figs. 2, 3, and 4 is 0.00001 just to be able to find exactly the value of λ_{max}. Table II provides these limits for Figs. 3 and 4. As shown in this table, $\lambda_{\max}$ is decreased when K increases for fixed values of T and β. On the other hand, λ_{max} increases as T decreases for fixed values of K and β. Lastly, λ_{max} increases when channel condition gets better for fixed values of T and K.

One important comment about the values of λ_{max}, when $T = 15$ and $K = 12$, which are 0.06829 and 0.06913 for β =1.7 and β =2.7 respectively. These values are the closest pair found in Table II. Does that mean that the values of the average buffer size around these values of λ_{max} do not differ much? We would like to state that although these values seems to be close to each other, it is necessary to know that when λ_{max} gets low, the average buffer size will differ with just a small increase of packet arriving probability (λ). For example, when $\lambda = 0.06826$ (a bit lower than λ_{max} for both cases), the average buffer size is about 897.7 and 37.1 for β =1.7 and β =2.7 respectively.

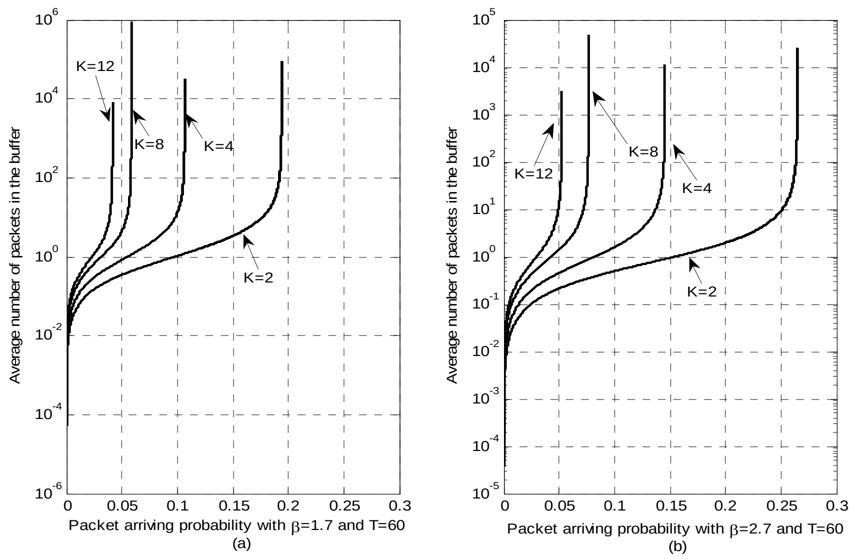

Figure 3: Average number of packets in the buffer versus packet arriving probability including fixed $T = 60$ and different working K and β.

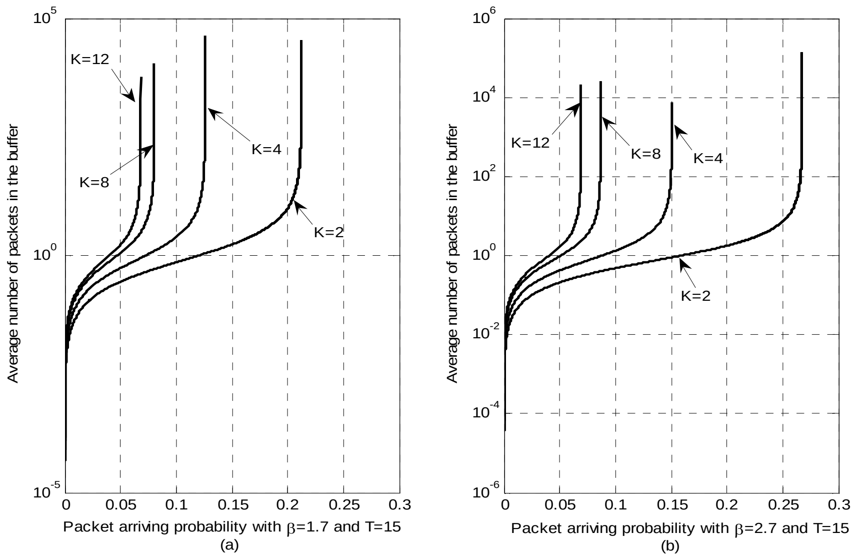

Figure 4: Average number of packets in the buffer versus packet arriving probability including fixed $T = 15$ and different working K and β.

TABLE I. AVERAGE NUMBER OF PACKETS IN THE BUFFER FOR VARIOUS SYSTEM PARAMETERS.

		$\beta = 1.7, T = 60$	$\beta = 2.7, T = 60$	$\beta = 1.7, T = 15$	$\beta = 2.7, T = 15$
Packet arriving probability (λ)	Lower bound decoding limit (K)	Average buffer size	Average buffer size	Average buffer size	Average buffer size
0.19330	2	365.3	1.744	6.304	1.591
0.10640	4	799.2	1.899	3.242	1.547
0.05813	8	196.3	2.186	1.616	1.318
0.04151	12	611.2	2.551	1.048	1.022

TABLE II. UPPER LIMIT OF PACKET ARRIVING PROBABILITY (λ_{max}) FOR VARIOUS SYSTEM PARAMETERS.

	$\beta = 1.7, T = 60$	$\beta = 2.7, T = 60$	$\beta = 1.7, T = 15$	$\beta = 2.7, T = 15$
Lower bound decoding limit (K)	λ_{max}	λ_{max}	λ_{max}	λ_{max}
2	0.19380	0.26470	0.21220	0.26730
4	0.10650	0.14530	0.12610	0.15050
8	0.05834	0.07646	0.08057	0.08678
12	0.04155	0.05232	0.06829	0.06913

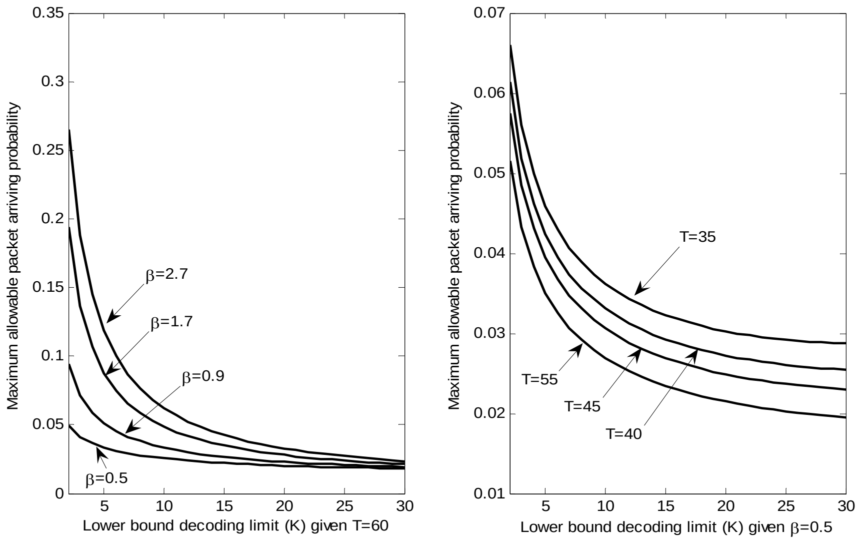

Figure 5: Maximum allowable packet arriving probability versus lower bound decoding limit (K) for different values of T and β.

To illustrate the case of existing λ_{max}, we can refer to load or system utilization which is defined in general [25] as the result of multiplying packet arrival rate with packet service rate for any system (server with its buffer). Also it is very important that the result of this product should be less than or equal to one in order to keep the buffer stable [14], [25]. In our proposed model, the system utilization is the packet arriving probability times the average decoding time, which is $\lambda\tau$. Hence, we conclude that there is λ_{max} which equals to $(1/\tau)$ for system stability. Consequently, when $\lambda > \lambda_{max}$, then the average queue size will go to infinity and this is what happens in Figs. 2, 3, and 4. The expression for average decoding time (τ) is found at (61).

Fig. 5 shows the maximum allowable packet arriving probability in order to preserve the stability of the system versus lower bound decoding limits. The decrease in λ_{max} is noticed when K increases. This indicates that the buffer gets larger. For a fixed value of K and T, the decrease in λ_{max} is seen as long as β decreases. Also, it is shown that λ_{max} is decreased, for a fixed value of K and β, when T increases. Hence, we can see that there is an effect of changing β and T on the buffer size by getting, as a result, various values of λ_{max}. Moreover, the results just explained of Fig. 5 (a) and (b) totally agree with our previous results and explanations about Figs. 2, 3, and 4. Table III is done to prove that through providing real numbers. It can be clearly seen that the results of this table, which are taken by applying the expression $1/\tau$, for λ_{max} are completely the same as the values found in Table II, which are obtained when applying our general form for the average buffer size, for λ_{max} with the same selected values of β, K, and T.

TABLE III.
MAXIMUM ALLOWABLE PACKET ARRIVING PROBABILITY FOR FIXED T AND DIFFERENT K AND β.

	$\beta = 1.7, T = 60$	$\beta = 2.7, T = 60$
Lower decoding limit (K)	λ_{max}	λ_{max}
2	0.19380	0.26470
4	0.10650	0.14530
8	0.05834	0.07646
12	0.04155	0.05232

IV. Simulation Results

The simulation of the previously modeled system is done through Matlab. A software-based approach is considered in order to validate our analytical observations. Throughout the duration of the simulation, Bernoulli RNG (random number generator) is invoked to simulate packets' arriving process as well as Pareto RNG which is programmed through utilizing the approach of inverse cumulative distribution function [27] to simulate packets' decoding time. As soon as a packet is recorded, the decoder starts decoding it at the beginning of the next time slot if it is not busy. The number of packets in the system buffer is managed until the duration of the simulation by defining a vector which is updated in every time slot where there is a possible packet arrival, a packet gets decoding, or both together at the same time slot. Consequently, the average number of packets can be found. However, Fig. 6 describes the average system buffer size obtained through simulation when employing different channel conditions, packet arriving probabilities, and upper bound decoding limits. The chosen value of lower bound decoding limit is the same as used in Fig. 2. The simulation time chosen is $4x10^5$ time slots. It is important to know that smooth results are obtained when choosing larger simulation time. This simulation has been run for four days with an increment of 0.01 for λ.

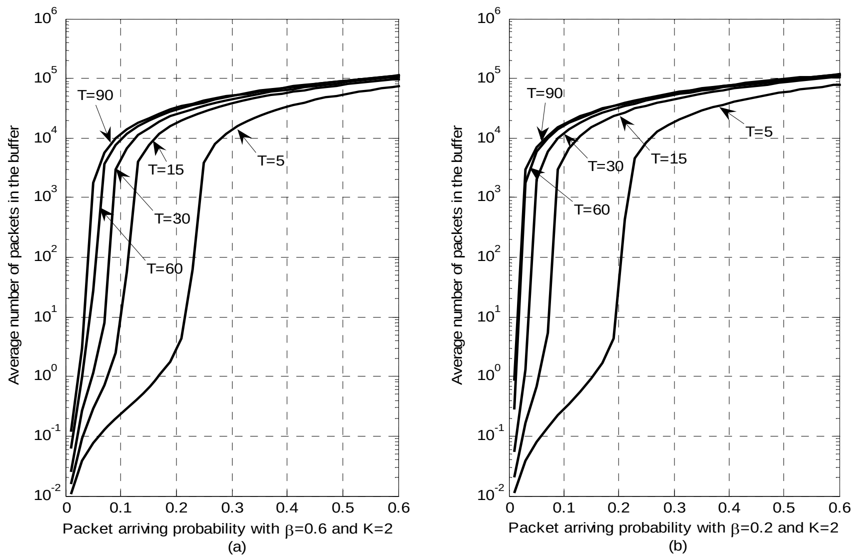

Figure 6: Average number of packets in the buffer versus packet arriving probability (λ) for different upper bound decoding limits and channel condition. Simulation time = $4x10^5$. (a) $\beta = 0.6$. (b) $\beta = 0.2$.

Actually, the simulation period is dependent on the increment selected for λ. The simulation has been performed on Intel Core 2 Duo 3.0 GHZ with a 2MB cache and 2 GB RAM. However, it is noticed that the average number of packets in the buffer increases as the packet arriving probability and the upper bound decoding limit increase. It is also seen an increase in the average number of packets as channel condition gets worse (i.e., β decreases).

One interesting observation can be made, which is the average number of waiting packets in the buffer reaches gradually to the working simulation time (i.e., maximum number of simulation slots) after being around a certain value of packet arriving probability which is mentioned in our analytical results as λ_{max}. In other words, the average buffer size moves gradually towards the duration of the simulation around $\lambda = 0.2$ for $(T = 5, \beta = 0.2)$ and $\lambda = 0.23$ for $(T = 5, \beta = 0.6)$. Also, it does so before $\lambda = 0.1$ for (T = 30, 60, and 90) when $\beta = 0.2$ and 0.6 with considering the notice that a faster move is the case when β decreases. While the case of T=15, the average buffer size moves around a limit after $\lambda = 0.1$ for $\beta = 0.6$ and before $\lambda = 0.1$ for $\beta = 0.2$. All of these simulation facts agree fairly with those shown in Fig. 2 as a result of theory.

The trend of moving gradually towards the duration of the simulation can be explained due to the finite number of simulation time slots $(4x10^5)$ where there is a maximum one packet that can arrive during any certain time slot. Hence, in the best case, a maximum of $4x10^5$ packets is the result. It is necessary to mention that if we increase the duration of the simulation (no matter what is the increase), the average buffer size will move gradually to that duration also around the mentioned limits of λ. This means reaching to infinity which verifies the same trend prescribed in theory as shown in Fig. 2.

V. CONCLUSION

In this paper, a hybrid type 1 ARQ with Fano decoding at the MAC layer of wireless access and end points is considered as an end-to-end improvement over wireless networks. A queuing analysis and simulation study are proposed for the system (queue and decoder) of wireless access and end points to obtain results about the expected number of packets in the system's buffer when Fano decoding or any other variable complexity decoding algorithms is implemented. This performance metric (average buffer size) has a severe impact on the wireless system performance and overall wireless network performance when it is chosen randomly. In the analytical study we derive a general form expression for the average size of the buffer that belongs to the Fano decoder, which is bounded by maximum and minimum variable decoding limits (T and K respectively), due to the unpredictable and noisy nature of wireless networks. This formulated expression is a function of not only (T and K), but also (β and λ). Analytical results show that the average buffer size increases dramatically when channel condition decreases (i.e., gets worse) by reaching λ_{max} which then goes to infinity. The effect of an increasing in λ on the average buffer size becomes less when β increases. Both variables T and K become highly sensitive to the increase of average buffer size when β decreases. Also, we provide results for a new derived form of λ_{max} that are totally the same as the λ_{max} results obtained through our general form expression of the average buffer size. On the other hand, we provide results obtained through simulation for the average buffer size. We show that our results and explanations for both analytical and simulation studies agree fairly with each other.

REFERENCES

[1] K.A. Darabkh and R. S. Aygün, “TCP Traffic Control Evaluation and Reduction over Wireless Networks Using Parallel Sequential Decoding Mechanism,” *EURASIP Journal on Wireless Communications and Networking*, vol. 2007, Article ID 52492 (SCI Expanded), 16 pages, 2007.

[2] Ye Tian, Kia Xu, and N. Ansari, “TCP in wireless environments: Problems and Solutions,” *Communications Magazine*, IEEE, Volume 43, Issue 3, pp. S27 - S32, Mar. 2005.

[3] M. Gast, *802.11 Wireless Networks: The Definitive Guide*, O'Reilly, 2005.

[4] C. Rinaldi, "Link-Layer Error Recovery Techniques to Improve TCP Performance over Wireless Links," *Master’s Thesis*, Royal Institute of Technology, Stockholm, Sweden, 2005.

[5] H. Balakrishnan and R. H. Katz, “Explicit loss notification and wireless web performance,” *Proc. of IEEE GLOBECOM '98,* Internet Mini-Conference, Sydney, Australia, Nov. 1998.

[6] Y.Wang, L.Pan, and J. Li, “The Necessity of Combining ELN and SACK to Improve TCP Performance over Heterogeneous Networks,” *Proc. of third International IEEE Conference on Signal-Image Technologies and Internet-Based System*, Shanghai, China, Dec. 2007, pp. 137-142.

[7] Y. S. Han, P.-N. Chen, and H.-B. Wu, "A maximum-likelihood soft-decision sequential decoding algorithm for binary convolutional codes," *IEEE Trans. on Commun.,* vol.50, no.2, pp.173-178, Feb. 2002.

[8] S. Lin, and D.J. Costello, *Error Control Coding: Fundamentals and Applications*, Pearson Education, 2004.

[9] J. B. Anderson, and S. Mohan, "Sequential Coding Algorithms: A Survey and Cost Analysis," *IEEE Trans. on Commun.,* vol. COM-32, no.2, pp.169-176, Feb. 1984.

[10] J. C. Moreira, and P. G. Farrell, *Essentials of Error-Control Coding*, John Wiley and Sons, 2006.

[11] R.O. Ozdag and P.A. Beerel, “A Channel Based Asynchronous Low Power High Performance Standard-Cell Based Sequential Decoder Implemented with QDI Templates,” *Proc. of 10th International Symposium on Asynchronous Circuits and Systems*, Crete, Greece, Apr. 2004, pp.187-197.

[12] K. A. Darabkh, and W. D. Pan, "Queuing Simulation for *Fano* Decoders with Finite Buffer Capacity," *Proc. of the 9th Communications and Networking Simulation Symposium (CNSS'06)*, Huntsville, Alabama, Apr. 2006.

[13] W. D. Pan, and A. Ortega, “Adaptive Computation Control of Variable Complexity Fano Decoders,” *IEEE Trans. on Commun.*, vol. 57, no. 6, pp. 1556 – 1559, June 2009.

[14] S. P. Meyn and R.L. Tweedie, *Markov Chains and Stochastic Stability*, Cambridge University Press, 2005.

[15] T. Hashimoto, "Bounds on a probability for the heavy tailed distribution and the probability of deficient decoding in sequential decoding," *IEEE Trans. on Inform. Theory*, vol. 51, issue 3, pp. 990 – 1002, Mar. 2005.

[16] R. Sundaresan and S. Verdu, “Sequential Decoding for the Exponential Server Timing Channel,” *IEEE Trans. on Commun.*, vol. 46, no. 2, pp. 705-709, March 2000.

[17] F. Jelinek, “An Upper Bound on Moments of Sequential Decoding Effort,” *IEEE Trans. on Inform. Theory*, vol. 15, no. 1, pp. 140-149, Jan. 1969.

[18] W.D. Pan and S.M. Yoo, "Queuing Analysis of sequential decoders with buffers," *Proc. of Huntsville Simulation Conference,* Huntsville, Alabama. Nov. 2004.

[19] W. D. Pan, and S.M. Yoo, "Fano Decoding with Timeout: Queuing Analysis," *ETRI Journal*, vol. 28, no. 3, pp. 301-310, June 2006.

[20] N. Shacham, "ARQ with Sequential Decoding of Packetized Data," *IEEE Trans. on Commun.,* vol. COM-32, pp. 1118-1127, Oct. 1984.

[21] K. A. Darabkh, and W.D. Pan, "Queuing Simulation for Sequential Decoders with Timeout," *Proc. of the 2005 Huntsville Simulation Conference*, Huntsville, Alabama, Nov. 2005.

[22] S. Kallel, and D. Haccoun, "Sequential decoding with an efficient partial retransmission ARQ strategy," *IEEE Trans. on Commun.,* vol. 39, Issue 2, pp. 208 – 213, Feb. 1991.

[23] K. A. Darabkh, and R. S. Aygun, "Performance Evaluation of Sequential Decoding System for UDP-Based Systems for Wireless Multimedia Networks," *Proc. of 2006 International Conference on Wireless Networks,* Las Vegas, Nevada, June 2006, pp. 365-371.

[24] K. Darabkh and B. Abu-Jaradeh, “Bounded Fano Decoders over Intermediate Hops Excluding Packet Retransmission,” *Proc. of IEEE 24th International Conference on Advanced Information Networking and Applications,* Perth, Australia, Apr. 2010.

[25] D. Cross, and C. M. Harris, *Fundamentals of Queuing Theory*, John Wiley and Sons, 1998.

[26] S. Halfin, "The shortest queue problem", *Journal of Applied Probability*, vol. 22, pp.865-878, 1985.

[27] J. Gentle, *Random Number Generation and Monte Carlo Methods (Statistics and Computing),* Springer, 2004.

Khalid A. Darabkh received his Ph.D. degree in Computer Engineering from University of Alabama in Huntsville, Alabama, USA in 2007. He is currently an assistant professor and the assistant dean for computer affairs in the Faculty of Engineering and Technology at the University of Jordan, Amman, Jordan. His main research interests include wireless and mobile communications, queuing theory, traffic management, multimedia systems and networking, congestion control architectures and resource allocation, parallel computing, and pattern recognition.