2010 Second International Conference on Network Applications, Protocols and Services

# Impact of Large Block FEC with Different Queue Sizes of Drop Tail and RED Queue Policy on Video Streaming Quality over Internet

Omar Almomani, Osman Ghazali, Suhaidi Hassan, Shahrudin Awang Nor and Mohammad Madi
Universiti Utara Malaysia, College of Arts and Sciences
Department of Computer Sciences
InterNetWorks Research Group
almomani81@yahoo.com,{osman,suhaidi,shah}@uum.edu.my, m_vip82@yahoo.com

**Abstract: In this paper, we report an investigation on the impact of large block Forward Error Correction (FEC) with Drop Tail (DT) and Random Early Detection (RED) queue policies on network performance and quality of video streaming. FEC is a technique that uses redundant packets to reconstruct dropped packets, while DT and RED are the most popular queue management policies used in network routers. DT mainly depends on the size of the queue buffer to decide on whether to drop a packet or not. RED monitors the average queue size and drops arriving packets probabilistically. The probability of dropping a packet increases as the estimated average queue size grows. In the investigation, we consider simulation settings with varying size of queue buffers. Results obtained from the simulation experiments show that large block FEC and queue size affect the performance the network. Consequently, the qualities of multimedia applications are also affected.**

## I. INTRODUCTION

Internet traffics suffer from heavy losses due to network congestion that is typically caused by the limited capacity of queues in the routers. A loss refers to a situation where a packet does not arrive at the destination, or arrive at the destination but late that caused it to be unusable. This usually happens when a network is heavily loaded. Congestion in the network is the most common reason for packet losses [1] [2]. This loss reduces the network performance. Therefore, packet loss and large delay in data transmission are often unacceptable.

Error control correction [3] [4] [5] is used to reconstruct the lost data by either retransmission of data from the sender using Automatic Repeat request (ARQ) or by adding redundant data using FEC. FEC [6] [7] is a method of error control correction, which is used to correct the error in data transmission by adding redundant data at the sender. When a receiver detects error in a packet, it will reconstruct the lost data from redundant data without retransmission of the lost data from the sender. However, there are several limitations of FEC. That is FEC cannot recover all lost packets. In addition, the transmission of redundant packets increases the overall network load [8]. The effectiveness of FEC is known to depend on the way packet drops are distributed in the data stream, i.e. dependent or independent packet drops. FEC is more efficient when packets losses are independent [9].

Queue policies refer to traffic policy techniques at a router that detect and notify traffic sources of imminent network congestion to prevent outbound buffer overflow and control queue delay [10]. When being notified of network congestion, cooperative traffic sources like TCP [11] [12] reduce their transmission rates in order to participate in the congestion control. In that case network congestion cannot be managed voluntarily by the traffic sources. Queue policies may use buffer management techniques to suppress traffic to the targeted traffic level and achieve the QoS goal. Traditional Internet routers used Drop Tail queue management [13], which drops the arriving packets if the buffer of the output port overflows. RED [14] solves the full queue drop packets by using the average queue size as the indication of emerging congestion.

This paper is organized in the following manner. Sections II present the description of FEC mechanism. Sections III present the description of queue policy. Section IV we present the simulation experiment setting. Section V we present the simulation results and discussion. Finally, Section VI concludes this paper.

## II. FEC MECHANISM

FEC has been proposed to recover packets loss in real time applications audio and video by using redundant information. A number of forward error correction techniques have been developed to repair data losses during transmission [15] [16] [17] [18]. FEC enables the receiver to correct losses without dealing with the sender.

FEC sends original and redundant data as a block of FEC (n, k), where $k$ is the number of data packets in a FEC block and $n$ is the number of all the packets in the FEC block. We can calculate the encoding rate using formula 1.

$$R = \frac{k}{n} \qquad 1$$

$R$ is the encoding rate of block. Codes that introduce less redundancy have higher code rates, and transmit more information per code bit.

There are two approaches to design FEC, i.e. media dependent or media-specific and media independent [19]. In the following subsection we explain each approach.

### A. MEDIA-INDEPENDENT FEC

Media- independent FEC [20] does not need to know what is inside the contents of the stream. Block or algebraic codes are transmitted to help repairing what was lost. There are $k$ data packets in a codeword and $n$-$k$ extra check packets are transmitted for $n$ packets that need to be sent over the

978-0-7695-4177-8/10 $26.00 © 2010 IEEE
DOI 10.1109/NETAPPS.2010.12

Internet. Figure 1 shows the way to use the media-dependent or media-specific FEC.

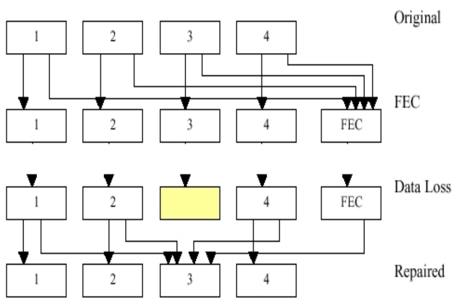


Figure 1:Media-independent FEC

### B. MEDIA-DEPENDENT FEC

Media dependent FEC [21] works against packet loss by transmitting each packet more than one time. When a packet is lost, one of its extra packets is able to restore it as shown in Figure .2. The first packet transmits the audio or video packet is the main encoding because it has the best quality. Duplicates of this packet is the minor encoding because the sender is able to decide if the quality or bandwidth of this packet should be the same or lower than the main encoding packet.

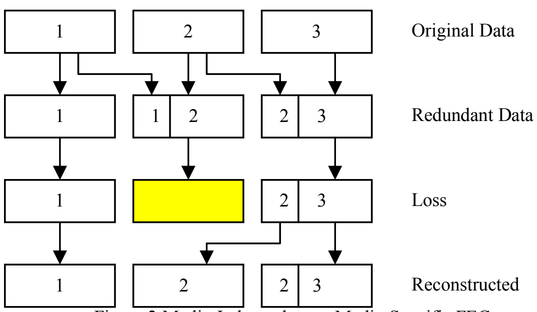


Figure 2:Media-Independent or Media-Specific FEC

## III. QUEUE POLICY

A queue policy monitors which of the arriving packets are dropped and which packets get in the queue. Once a packet gets in the queue it is ultimately sent. The purpose of a queue policy is to maximize the sum of the benefits of all the packets it delivers. Traditional Internet routers employ DT queue management, discarding arriving packets if the buffer of the output port overflows. A new queue management policy proposed in [14], called RED, was to improve the Tail Drop policy. In the following sub-section we explain each policy.

### A. Drop Tail

DT is a simple queue policy algorithm used by Internet routers to decide when to drop packets. In contrast to the more complex algorithms like RED [14], DT does not differentiate packets that means each packet is treated equally [22]. With DT, when the queue is filled to its maximum capacity, the newly arriving packets will be dropped until the queue has enough spaces to accept the incoming traffic. The DT has two disadvantages:

i. Lock-out.
ii. Full Queue.

The solution to the full-queues problem is for routers to drop packets before a queue becomes full, so that end nodes can respond to congestion before buffers overflow. We call it a proactive approach. By dropping packets before buffers overflow, active queue management allows routers to control when and how many packets to be dropped. Figure 3 shows Drop Tail's infrastructure.

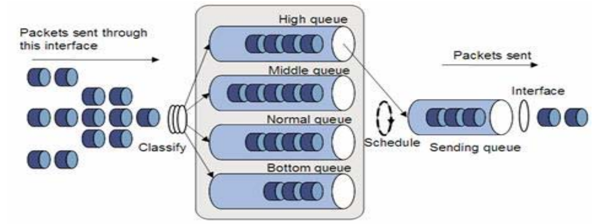


Figure 3: Drop Tail's infrastructure[1]

### B. RED

RED [14] is the most studied active queue policy mechanism on the Internet. RED solves the full queue drop packets problem by using the average queue size as the indication of emerging congestion. The RED algorithm drops arriving packets probabilistically. The probability of drop increases as the estimated average queue size grows. If the queue has been mostly empty in the recent past, RED will not tend to drop packets unless the queue overflows. On the other hand, if the queue has recently been relatively full, indicating persistent congestion, newly arriving packets are more likely to be dropped. The RED algorithm consists of two main parts: [23]

i. Estimation of the average queue size - RED estimates the average queue size, either in the forwarding path using a simple exponentially weighted moving average.
ii. For the decision to drop an incoming packet, RED decides whether or not to drop an incoming packet. It is RED's particular algorithm for dropping those results in performance improvement for responsive flows. There are two RED parameters, minth (minimum threshold) and maxth (maximum threshold).

RED effectively controls the average queue size while still accommodating bursts of packets without loss. RED's use of randomness breaks up synchronized processes that could lead to lock-out phenomena. Figure 4 shows the RED's infrastructure.

[1]http://www.h3c.com/portal/Products___Solutions/Technology/QoS/Technology_Introduction

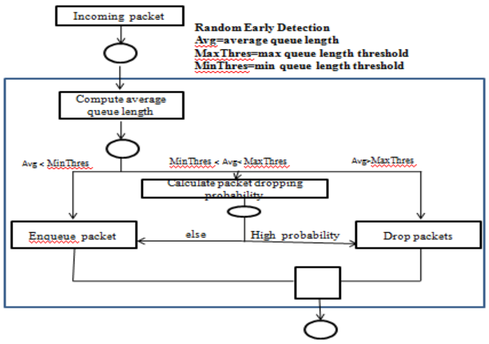


Figure 4:RED's infrastructure[2]

## IV. SIMULATION SETTINGS

To study the effects of different buffer size of the DT and RED queue policy with large block FEC on network performance and video quality, we conducted a simulation for single bottleneck topology (dumbbell) as show in Figure 5. We used the dumbbell topology because it is the most suitable topology for our research [23], This topology is similar to other topologies used by other works [24] [25].

More than 80%of Internet traffic today consists of TCP traffic; we used competing TCP traffic flow to increase the packet losses. The TCP sources connected to R1 with a bandwidth of 10Mbps and 5ms delay, the FTP traffic attached to TCP sources. The FEC sources connected to R1 with a bandwidth of 10Mbps and delay was generated randomly using uniform distribution to achieve the heterogeneous environment of Internet. We used the Constant Bit Rate (CBR) model in ns-2 for video traffic because it closely represents the behavior of real video data, and attached it to FEC.

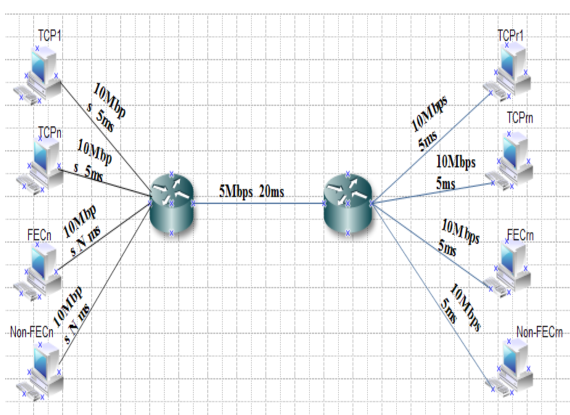


Figure 5:Simulation Topology

Router 1 was connected with Router 2 with a bandwidth of 5Mbps and a 20ms delay link. These configurations cause congestion and loss at R1 so we can identify the efficiency of the FEC to recover packet losses based on received packets at the receiver. The router in this experiment used the DT and RED queue policy management,

The simulation was run for 30 times. A random number generator is used to randomly generate the starting time of traffic flow. The experiment was run for 100 seconds. The first 20 seconds was ignored due to instability of the simulation in initial start up. The results were presented with a 95% confidence interval.

## V. RESULTS AND DISCUSSION

Tables 1 and 2 present the results of simulation experiment. Table 1 shows the results of FEC with RED and Table 2 shows the results of FEC with Drop Tail.

Table 1. FEC with RED

| Queue Size | Total Loss | Total Received | Bandwidth Kbps | Delay ms |
|---|---|---|---|---|
| 20 | 245±6 | 6895 | 1975±2 | 499±101 |
| 40 | 241±4 | 6899 | 1976±1 | 503±106 |
| 60 | 240±1 | 6900 | 1977±0.5 | 510±181 |
| 80 | 236±5 | 6894 | 1977±2 | 530±116 |
| 100 | 238±9 | 6892 | 1977±3 | 553±115 |

Table 2. FEC with Drop Tail

| Queue Size | Total Loss | Total Received | Bandwidth Kbps | Delay Ms |
|---|---|---|---|---|
| 20 | 237±11 | 6903 | 1977±3.17 | 518±116 |
| 40 | 187±18 | 6953 | 1992±5.21 | 534±123 |
| 60 | 156±9 | 6984 | 2001±2.64 | 551±113 |
| 80 | 117±2 | 7023 | 2012±0.82 | 609±121 |
| 100 | 96±9 | 7044 | 2018±2.72 | 647±108 |

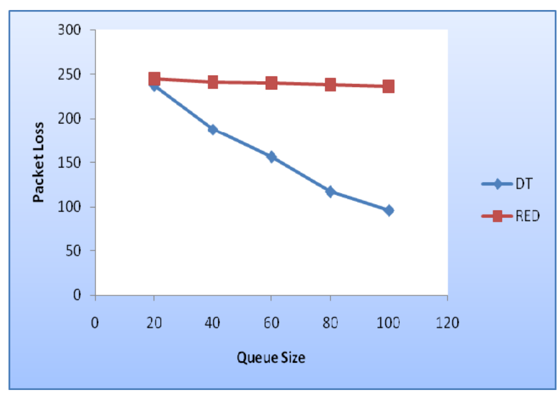


Figure 6:Packet with DT and RED

Figure 6 shows the packet loss with DT and RED. We can observe that when queue size increased the DT performed better than RED. As we mentioned, the DT just depends on the queue size parameter to drop the packet. Therefore, when buffer size increase more packets can enter the queue then fewer packets will be lost. While in RED beside queue size there are four other parameters used in deciding to drop a packet. Since the RED drops arriving packets probabilistically. Therefore, the buffer size on RED doesn't affect the number of packet loss as much as TD.

[2]http://commons.wikimedia.org/wiki/File:Random_Early_Detection_algorithm_en.svg

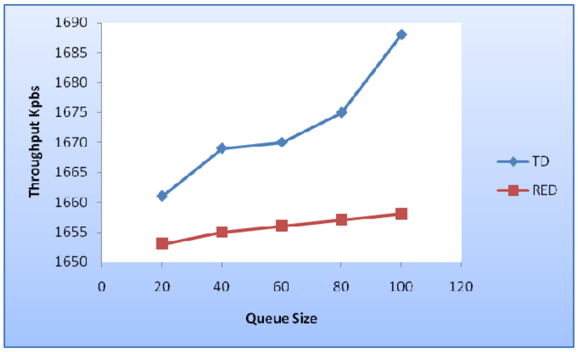


Figure 7:Throughput with DT and RED

Figure 7 shows the throughput with DT and RED. From the figure we can observe that the DT produced more throughput than RED with an increase in queue size. This is because the large DT queue size has less packet loss therefore more packets received by the receiver.

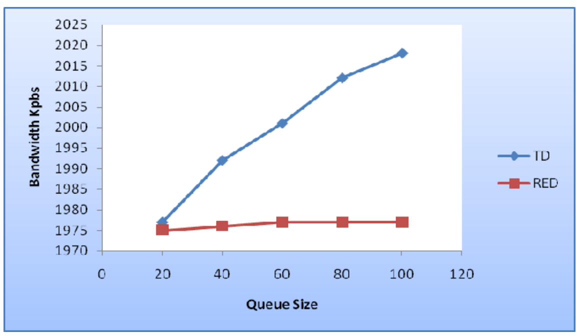


Figure 8:Bandwidth with DT and RED

Figure 8 shows the bandwidth usage with DT and RED. From the figure we can observe that the DT requires more bandwidth than RED with an increased queue size. This because the large DT queue size has less packet loss so more packets would be transferred.

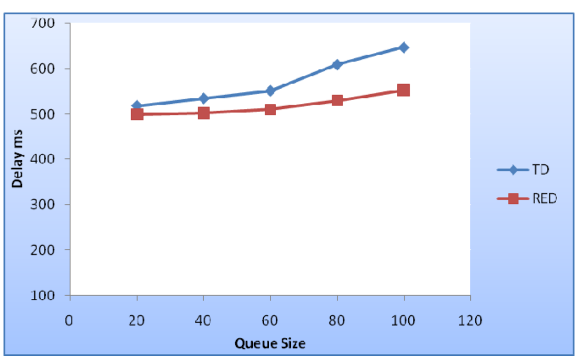


Figure 9:End-to-end delay with DT and RED

Figure 9 shows the end-to-end delay with DT and RED. From the figure we can observe that the DT produced more delay than RED with the increased queue size. This is because the packets have to wait longer in the queue before they are forwarded from the router to the network.

## VI. CONCLUSION

In this paper, we examined the current FEC mechanism with the most popular queue policy use in today Internet routers, i.e. DT and RED with varying queue sizes, conducted using a set of simulation experiments. Our results showed that the RED performed better than DT from a network perspective, since the RED required less bandwidth and end-to-end delay with various queue sizes. While DT performed better than RED from the quality perspective, since the DT had less packet loss and more throughputs with various queue sizes.

## REFERENCES


[1] I. Cidon, A. Khamisy, and M. Sidi, "Analysis of Packet Loss Processes in High-Speed Networks" *IEEE Transactions on Information Theory*, vol. 39, pp. 98-108, 1993.

[2] S.-R. Kang and D. Loguinov, "Modeling Best-Effort and FEC Streaming of Scalable Video in Lossy Network Channels" *Networking, IEEE/ACM Transactions on*, vol. 15, pp. 187-200, 2007.

[3] Robert.H and Morelos-Zaragoza, *The Art of Error Correcting Coding*, Second ed. England: John Wiley & Sons Ltd, 2006.

[4] K. R. Rao, Zoran S. Bojkovic, and D. A. Milovanovic, *Introduction To Multimedia Communications Applications, Middleware, Networking*. New Jersey, USA: John Wiley & Sons, Inc., 2006.

[5] T. Moon, *Error Correction Coding: Mathematical Methods and Algorithms*. Hoboken, New Jersey,USA: Wiley-Interscience, 2005.

[6] J. Lacan, V. Roca, J. Peltotalo, and S. Peltotalo, "Reed-solomon forward error correction (FEC) schemes" Proceedings of, 2009 , http://www.ietf.org/rfc/rfc5510.txt?number=5510.

[7] Y. Wang and Q.-F. Zhu, "Error Control and Concealment for Video Communication: A Review" *Proceedings of the IEEE*, vol. 86, pp. 974 - 997, May 1998.

[8] S.-r. Kang and D. Loguinov, "Impact of FEC Overhead on Scalable Video Streaming," in *Proceedings of the international workshop on Network and operating systems support for digital audio and video*. Stevenson, Washington, USA: ACM, 2005, pp. 123-128.

[9] T. Alemu, Y. Calas, and A. Jean-Marie, "The Interaction of Forward Error Correction and Active Queue Management" Proceedings of the IFIP Networking'05, Waterloo, Ontario, Canada, 2005.

[10] B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, and C. Partridge, "RFC2309: Recommendations on Queue Management and Congestion Avoidance in the Internet" *Internet RFCs*, 1998.

[11] V. Jacobson, R. Braden, and D. Borman, "TCP extensions for high performance," RFC 1323, May 1992. http://www.ietf.org/rfc/rfc1323.txt?number=1323.

[12] T. Socolofsky and C. Kale, "A TCP/IP Tutorial" *RFC 1180, The Internet Engineering Task Force*, 1991. http://www.ietf.org/rfc/rfc1180.txt.

[13] C. Brandauer, G. Iannaccone, C. Diot, T. Ziegler, S. Fdida, M. May, and S. Res, "Comparison of Tail Drop and Active Queue Management Performance for Bulk-data and Web-like Internet Traffic" Proceedings of the Sixth IEEE Symposium on Computers and Communications, 2001.

[14] S. Floyd and V. Jacobson, "Random Rarly Detection Gateways for Congestion Avoidance" *IEEE/ACM Transactions on networking*, vol. 1, pp. 397-413, 1993.

[15] P. Frossard, "FEC Performance in Multimedia Streaming" *IEEE Communications Letters*, vol. 5, pp. 122-124, 2001.

[16] M. Podolsky, C. Romer, and S. McCanne, "Simulation of FEC-Based Error Control for Packet Audio on the Internet" Proceedings of the IEEE INFOCOM '98, Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies, San Francisco, CA, USA, 1998.

[17] Q. Zhao, C. Li, X. Zhang, and D. Liu, "A New FEC Scheme for Real-time Transmission of High Definition Video over IP Network" Proceedings of the The International Conference on Intelligent Pervasive Computing, Jeju Island, Korea, 2007.

[18] J. Cai and C. W. Chen, "FEC-Based Video Streaming over Packet Loss Networks with Pre-Interleaving" Proceedings of the International Conference on Information Technology: Coding and Computing, Las Vegas, NV USA, 2001.

[19] X. Chen, C. Wang, D. Xuan, Z. Li, Y. Min, and W. Zhao, "Survey on QoS Management of VoIP" *Proceedings of the 2003 International Conference on Computer Networks and Mobile Computing*, pp. 69, 2003.

[20] C. Perkins, O. Hodson, and V. Hardman, "A Survey of Packet Loss Recovery Techniques for Streaming Audio" *IEEE Network*, vol. 12, pp. 40-48, 1998.

[21] G. Rubino and M. Varela, "Evaluating the Utility of Media-Dependent FEC in VoIP Flows" *LECTURE NOTES IN COMPUTER SCIENCE*, pp. 31-43, 2004.

[22] G. Nguengang, D. Gaiti, and M. de Castro, "A Differentiated Services API for Adaptive QoS Management in IP Networks" *Proceedings of Fourth IFIP International Conference on Network Control and Engineering for QoS, Security, and Mobility, Lannion, France, November 14-18, 2005*, pp. 117, 2007.

[23] B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, and C. Partridge, "Recommendations on Queue Management and Congestion Avoidance in the Internet" *Internet Engineering Task Force,RFC2309*, 1998.http://www.ietf.org/rfc/rfc2309.txt.

[24] R. Rejaie, M. Handley, and D. Estrin, "RAP: An End-to-End Rate-Based Congestion Control Mechanism for Realtime Streams in the Internet" Proceedings of the Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies, New York, NY USA, 1999.

[25] K. French and M. Claypool, "Repair of Streaming Multimedia with Adaptive Forward Error Correction" Proceedings of the ITCOM, Denver, CO, USA, 2001.