Improving Reliable Transmission Throughput with Systematic Random Code

Zan-Kai Chong*, Hiroyuki Ohsaki†, Bryan Ng‡, Bok-Min Goi*, Hong-Tat Ewe*, Sin-Ran Chong*
* Lee Kong Chian Faculty of Engineering and Science, Universiti Tunku Abdul Rahman, Selangor, Malaysia.
† School of Science and Technology, Kwansei Gakuin University, Japan.
‡ Faculty of Engineering, Victoria University of Wellington, New Zealand.

***Abstract*—Rateless erasure code (REC) is an erasure code, where the encoder generates a potentially infinite number of encoded symbols and the original message can be reconstructed from a sufficient number of correctly received packets. Many REC-based transmission protocols have been proposed for improving network throughput in lossy channel. However, state-of-the-art RECs (such as LT code and Raptor code) are not efficient for transmitting short messages. Recent studies suggest that network traffic is characterised by bursts of short messages and thus existing transmission protocols do not benefit from the gains of deploying REC. In this paper, we propose an REC-based transmission protocol, namely UDP-RC, which integrates the simplicity of UDP and strength of systematic Random code suited to network traffic with short messages. It attains high throughput by transmitting short messages reliably with lower overheads over lossy channel. We experimentally show that UDP-RC achieves at least 50% higher throughput and maintains more stable throughput compared to TCP (Transmission Control Protocol) and UDT (UDP Data transfer) protocol under both ideal and lossy channel conditions.**

I. Introduction

Erasure code is an error correction code (also known as forward error correction code (FEC)) for erasure channel. An erasure channel models the probability a bit is received in error with a constant probability ρ. Generally, a message of k symbols is encoded into n encoded symbols and the original message can be reconstructed from any k out of n encoded symbols. A promising application of erasure code is in network coding, and to that effect erasure code has been employed in several variants of transmission protocols such as TCP to improve the network throughput [1], [2], [3].

Near optimal erasure codes in networking are typically realised as a *rateless* erasure code (REC). An REC system encodes a message of k symbols into a potentially *infinite* number of encoded symbols. Unlike typical erasure codes, REC reconstructs the original message from any $k\,(1+\epsilon)$ encoded symbols with high success probability (e.g., 99.9%), where ϵ denotes the decoding inefficiency. For example, the state-of-the-art Raptor code achieves a low decoding inefficiency of ($\epsilon = 0.03$) when transmitting a long message ($k = 100,000$ symbols) [4].

Transmission protocols (also known as transport layer protocols) with REC are desirable for two reasons. Firstly, some congestion control functions are replicated in transmission protocols, which may already be offered at the link layer and thus adding complexity to the protocol stack. Secondly, transmission protocols with REC reduces buffering requirements and in the future may allow: (i) certain links to do away with buffering altogether thus reducing buffer bloat problems [5] (ii) efficient data transfer over sensor networks, for example the structural health monitoring scenario described in [6].

The deployment of REC improves the network throughput by (i) minimising the retransmission of lost packets and (ii) reducing congestion control overheads. Taking this idea one step further, REC paves the way for a network that does not require congestion control mechanisms at end hosts. Raghavan and Snoeren conjectured the existence of a perfect REC (i.e., $\epsilon = 0.0$) whereby all senders may transmit messages at the maximum speed of the available channel capacity [7]. Once the sender and receiver pace reaches an equilibrium, every sender will have fair use of bandwidth with no congestion collapse. Such ideas have been analysed in [8], [9], [10] and many REC-based transmission protocols such as FECTCP [11], [12] and Digital Fountain based Communication Protocol (DFCP) [10], [13] have been proposed.

A caveat in REC for networking protocols is that it is efficient for long messages. However, recent studies of several bandwidth-intensive applications such as high definition video streaming and exascale data transfer, have challenged the common belief that long messages contribute to the majority of the network traffic. According to traffic characteristics published in [14], 80% of the flow in data centres are smaller than 10KB in size. Analysis of Internet flow in [15] also shows that 80% of China Education and Research Network (CERNET) flow have fewer than ten packets and 90% of flow have size less than 7KB. The findings in CERNET is also consistent with traffic traces from Center for Applied Internet Data Analysis (CAIDA) that shows that 80% of flow packets have fewer than five packets and 90% of flow have size less than 2KB.

The analysis of network traces clearly shows that short messages make up the majority of the network traffic. Because of this, the aforementioned REC-based transmission protocols will be of little benefit to current networks because transport protocols using REC are not efficient for transmitting short messages. To realise the gains from RECs, we propose a new transmission protocol for short messages with the benefits of REC called UDP-RC. It integrates the simplicity of User Data-

DOI 10.1109/LCN.2016.87

gram Protocol (UDP) and the strength of systematic Random code. UDP is widely used in real time transmission, where lost packets are not retransmitted and congestion avoidance algorithm are unnecessary. Systematic Random code is a REC that is efficient for short messages [16]. It reconstructs the original message if the first k encoded symbols are received intact. Otherwise, $k + 10$ encoded symbols (i.e., ten overhead symbols) are required in order to reconstruct original message with 99.9% success probability irrespective of the message length, k.

The proposed UDP-RC embraces the idea of Raghavan and Snoeren[7] and does not employ congestion avoidance algorithm and retransmission mechanism for lost packets. In brief, the sender will transmit the first $k + 10$ packets at a maximum speed that is controlled by the application and the rest of the packets will be sent at a controlled transmission rate. Unlike other papers,we evaluate the new UDP-RC through test bed experiments and we demonstrate that it achieves higher and more stable throughput compared to TCP and UDT (a UDP-based reliable transmission protocol [17]).

The remainder of this paper is organised as follows. UDP-RC is proposed in Section II, which explains the packet exchange sequence between sender and receiver and how reliable transmission is achieved without retransmission and congestion avoidance. Section III presents the experimental results over production networks comparing UDP-RC with TCP and UDT in terms of throughput and time to transfer files of fixed sizes. Finally, the conclusion is presented in Section IV.

II. High Throughput Transmission Protocol

In this section, we describe the REC-based transmission protocol, namely UDP-RC and the packet transmission and acknowledgement mechanisms are elaborated in Figure 1. The source code for UDP-RC is publicly available at [18].

Generally, UDP-RC consists of two types of packets - data (DATA) and acknowledge (ACK) packets. The sender sends the data packets that contain the fields of data id (DATA_ID), encoded data (DATA) with the necessary information for decoding (e.g., message length, symbol size, etc.) and exit flag (EXIT) to the receiver. The receiver will response each data packet with an acknowledgement packet that contains the acknowledgement id (ACK_ID) and the exit flag.

We illustrate the transmission and acknowledgement mechanisms by referring to the packets flow diagram in Figure 1. Unlike other reliable transmission protocols, UDP-RC does not establish the connection explicitly (e.g., TCP uses a SYN packet to establish connection). It starts the transmission with the first data packet (i.e., DATA_ID=1) that contains the encoded data (DATA) (see Figure 1(a)). If no acknowledgement packet is received within the timeout period, the connection establishment is said to be unsuccessfull. Meanwhile, the arrival of the first acknowledgement packet implies the connection establishment.

In this paper, UDP-RC uses a simple transmission rate control mechanism with no congestion avoidance algorithm.

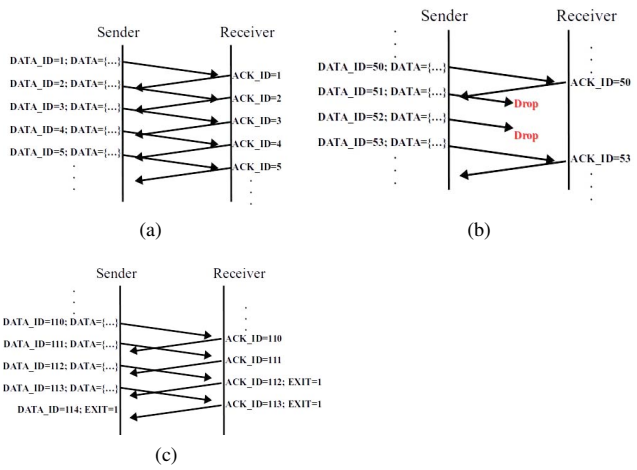


Figure 1: Packet flow diagrams of UDP-RC for (a) establishing connection, (b) encountering packet losses, (c) closing connection.

That is, after sending the data packet, the sender will sleep for t (milli) second and then continue for the next data packet. In particular, the sender will not slow down (by changing the value of t) the transmission rate even though there is missing acknowledgement Figure 1(b)). Tuning the parameter t is not investigated in this paper, but we note that this represents an opportunity for further optimising UDP-RC.

On the receiving end, the receiver acknowledges every received data packet with acknowledgement packet of ACK_ID = DATA_ID (see Figure1(b)). These packets signal the sender to disconnect the connection. Once the receiver has sufficient packets to decode (i.e., $k+10$), the acknowledgement packet is sent with the exit flag, i.e., EXIT $= 1$. Upon receiving this packet, the sender closes the connection by sending an empty packet (i.e., without any encoded data) with the exit flag to acknowledge the termination (see Figure 1(c)).

III. Experimental Result

A set of experiments is conducted over two different types of links to demonstrate the performance of the UDP-RC. We conduct experiments over best effort Internet between Universiti Tunku Abdul Rahman (UTAR) in Malaysia and Victoria University of Wellington (VUW) in New Zealand and Kyoto University in Japan using the ASPEN testbed [19]. The next set of experiments are conducted over a microwave point-to-point link connecting two points (11.3km apart) over Porirua in New Zealand. These experiments reflect the typical usage scenarios for network coding and provide an interesting context to compare how UDP-RC performs under different capacity and erasure probabilities. Note that the comparison in between UDP-RC and UDT is potentially unfair as the former is written in Python while UDT in C++. In particular, UDP-RC is not yet optimised for best computing performance.

We do not compare UDP-RC with other REC transmission protocols because their source code are not available

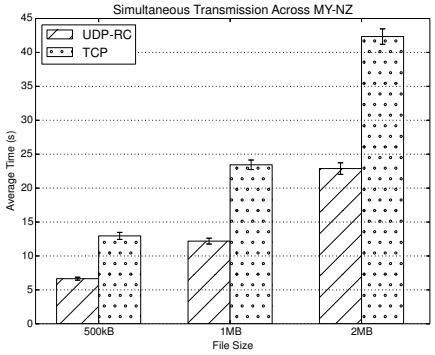


Figure 2: Average transfer time in seconds for TCP and UDP-RC. Connection between UTAR and VUW over best effort Internet.

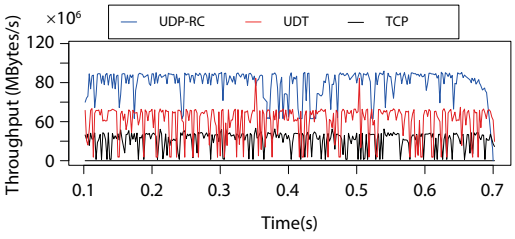
online and their inappropriate assumption on the efficiency of their employed state-of-the-art RECs in transmitting short messages. We are aware of LT code variants that are dedicated for short messages such as [20], [21], but they require more overhead symbols to reconstruct the original messages with high success probability. Interested readers are referred to our earlier paper [16] for the details.

A. *Best effort Internet*

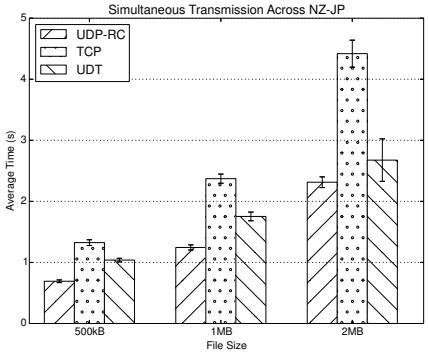
We use two desktop computers (running Linux) to function as end points and these end points are connected over the Internet. The tests are conducted over two pairs of end points: (i) between UTAR, Malaysia and VUW, New Zealand and (ii) between VUW and Kyoto University, Japan. The end point in UTAR has a 1Mbps uplink and 4Mbps downlink Asymetric Digital Subscriber Line (ADSL) connection to the Internet, while both VUW and Kyoto University end points have a 1Gbps uplink and 1Gbps downlink switched Ethernet connection to the campus network.

In these experiments, the channel (or more precisely the path) between end points is established via IP (Internet Protocol). The channel capacity is assumed to vary during the entire duration of the experiment. Such capacity variations were observed during our experiments and are expected for best effort Internet. We calculate the erasure probability by comparing bit-level logs from the Linux kernel at both sender and receiver to provide some context to the performance comparison.

The average transfer time for a file of size 500KB, 1MB and 2MB is plotted in Figure 2. Results from the test between end points in UTAR and VUW shown in Figure 2 reveal that UDP-RC reduces the average transfer time by almost half compared to TCP and this observation is consistent across different file sizes of 500KB, 1MB and 2MB. We were unable to conduct experiments with UDT between UTAR and VUW due to some restrictions imposed by the network service provider at the UTAR end point.



(a)



(b)

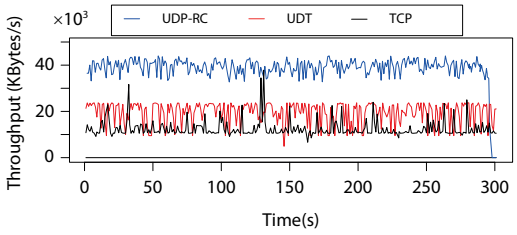
Figure 3: (a) Instantaneous throughput for TCP, UDT and UDP-RC, (b) average transfer time in miliseconds. Connection between VUW and Kyoto University over best effort Internet with average erasure probability of 0.013%.

Results from the test between end points in VUW and Kyoto University is shown in Figure 3. The results in Figure 3(a) shows the instantaneous throughput for transmitting 600 files (each 100KB) from VUW to Kyoto and we observe that UDP-RC achieves 73% and 50% higher throughput respectively (around 85MBps) than both TCP and UDT. Throughput for UDP-RC clearly tapers off faster than both TCP and UDT indicating that the transfer time for a fixed size file is shorter for UDP-RC.

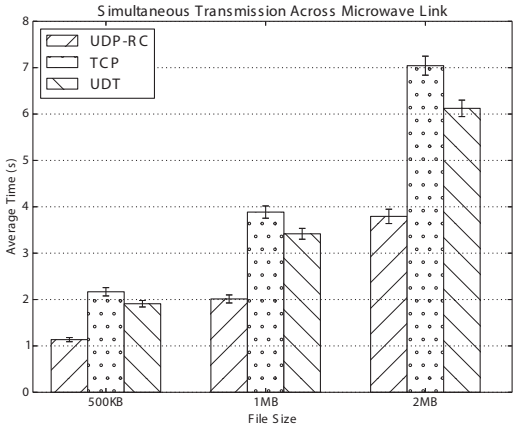
Further investigation of the transfer time is shown in Figure 3(b). The average transfer time for a file of size 500KB, 1MB and 2MB is plotted in Figure 3(b) and these results are obtained with average erasure probability 0.013% for this path. As expected the average transfer time for UDP-RC is the shortest at 7ms while TCP records the highest time to complete the file transfer at 13ms (for file size of 500KB). Moreover, the standard deviation for UDP-RC is significantly smaller than both TCP and UDT indicating low variability. These observations are similar across different file sizes of 1MB and 2MB.

B. *Wireless microwave point-to-point*

Wireless microwave point-to-point links are commonly used to carry aggregated traffic from mobile base stations in 2G/3G (second and third generation) wireless cellular communications, eNodeB (fourth generation cellular wireless systems),



(a)



(b)

Figure 4: (a) Instantaneous throughput for TCP, UDT and UDP-RC, (b) average transmission time in seconds. Microwave connection in Wellington with average erasure probability of 1.06%.

and communication between earth stations for satellite systems. We used a point-to-point link over the 22.022GHz licensed band with transmit power of 10dBW and link capacity of 2Mbps across Porirua in Wellington, New Zealand. Weather conditions have a strong influence on microwave link loss, the experiments were conducted on the 22nd April 2016, and the weather conditions in Wellington for the date the experiments were carried out are available at niwa.co.nz.

Results from the test across the wireless microwave link is shown in Figure 4. The results again show that UDP-RC (~43KBps) achieves higher throughput than both TCP (~22KBps) and UDT (~26KBps) for transferring 600 files each 100KB. The average transfer time for a file of size 10kB, 1MB and 2MB is plotted in Figure 4(b) and these results are obtained with average erasure probability 1.06% for this path. The average transfer time for UDP-RC is the shortest at 1.2s while TCP records the highest time to complete the file transfer at 1.9s. This observation is similar across different file sizes of 1MB and 2MB.

IV. Conclusion

In this paper, we proposed an REC-based transmission protocol, namely UDP-RC, where it integrates the simplicity of UDP and strengths of systematic Random code. The UDP-RC utilises the full potential of REC, where a high transmission throughput is achieved without congestion avoidance algorithm and retransmission of lost packets. We experimentally show that UDP-RC achieves higher and more stable throughput compared with TCP and UDT. In the absence of congestion avoidance algorithm, UDP-RC potentially suffers from two issues: (i) unfainess to other TCP flow in terms of bandwidth utilisation, and (ii) frequent packet bursts for sustained durations will overwhelm the receiver. A better flow control mechanism will be explored in the future work.

References

[1] T. Tsugawa, N. Fujita, T. Hama, H. Shimonishi, and T. Murase, "TCP-AFEC: An adaptive FEC code control for end-to-end bandwidth guarantee," in *Packet Video 2007*. IEEE, 2007, pp. 294–301.

[2] B. Ganguly, B. Holzbauer, K. Kar, and K. Battle, "Loss-tolerant TCP (LT-TCP): Implementation and experimental evaluation," in *Military Communications Conference (MILCOM)*. IEEE, 2012, pp. 1–6.

[3] K. Krishnaprasad, M. P. Tahiliani, and V. Kumar, "TCP kay: An end-to-end improvement to TCP performance in lossy wireless networks using ACK-DIV technique & FEC," in *CONECCT*. IEEE, 2015, pp. 1–6.

[4] A. Shokrollahi, "Raptor codes," *IEEE/ACM Trans. Netw.*, vol. 14, no. SI, pp. 2551–2567, Jun. 2006.

[5] J. Gettys and K. Nichols, "Bufferbloat: Dark buffers in the Internet," *Queue*, vol. 9, no. 11, p. 40, 2011.

[6] S. Singh, W. K. Seah, and B. Ng, "Cluster-centric MAC for WSNs in structural health monitoring," in *Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks*. IEEE, 2015, pp. 275–282.

[7] B. Raghavan and A. C. Snoeren, "Decongestion control," in *Fifth Workshop on Hot Topics in Networks*, 2006, pp. 61–66.

[8] T. Bonald, M. Feuillet, and A. Proutiere, "Is the 'law of the jungle' sustainable for the Internet?" in *INFOCOM 2009, IEEE*, april 2009, pp. 28 –36.

[9] L. López, A. Fernández, and V. Cholvi, "A game theoretic comparison of tcp and digital fountain based protocols," *Computer Networks*, vol. 51, no. 12, pp. 3413–3426, 2007.

[10] S. Molnar, Z. Móczár, and B. Sonkoly, "How to transfer flows efficiently via the Internet?" in *ICNC*. IEEE, 2014, pp. 462–466.

[11] A. Botos, Z. Polgar, V. Bota *et al.*, "Analysis of a transport protocol based on rateless erasure correcting codes," in *International Conference on Intelligent Computer Communication and Processing (ICCP)*. IEEE, 2010, pp. 465–471.

[12] A. Botos, Z. Polgar, Z. Kiss *et al.*, "FECTCP for high packet error rate wireless channels," in *International Conference on Communications (COMM)*. IEEE, 2010, pp. 327–330.

[13] S. Molnar, Z. Móczár, A. Temesváry, B. Sonkoly, S. Solymos, and T. Csicsics, "Data transfer paradigms for future networks: Fountain coding or congestion control?" in *IFIP Networking Conference, 2013*. IEEE, 2013, pp. 1–9.

[14] T. Benson, A. Akella, and D. A. Maltz, "Network traffic characteristics of data centers in the wild," in *ACM SIGCOMM*. ACM, 2010, pp. 267–280.

[15] X. Zhang and W. Ding, "Comparative research on Internet flows characteristics," in *International Conference on Networking and Distributed Computing (ICNDC)*. IEEE, 2012, pp. 114–118.

[16] Z.-K. Chong, B.-M. Goi, H. Ohsaki, B. C.-K. Ng, and H.-T. Ewe, "Systematic rateless erasure code for short messages transmission," *Computers & Electrical Engineering*, vol. 45, pp. 55–67, 2015.

[17] Y. Gu and R. L. Grossman, "UDT: UDP-based data transfer for high-speed wide area networks," *Computer Networks*, vol. 51, no. 7, pp. 1777–1799, 2007.

[18] Z.-K. Chong, "User datagram protocol with rateless erasure code UDP-RC," https://github.com/zkchong/UDP-RC, 2016.

[19] B. Ng, M. Hayes, and W. K. Seah, "Developing a traffic classification platform for enterprise networks with SDN: Experiences & lessons learned," in *IFIP Networking Conference*. IEEE, 2015, pp. 1–9.

[20] J. K. Zao, M. Hornansky, and P. lun Diao, "Design of optimal short-length LT codes using evolution strategies," in *Proc. IEEE Congress Evolutionary Computation (CEC)*, 2012, pp. 1–9.

[21] W. Zhang and S. Hranilovic, "Short-length raptor codes for mobile free-space optical channels," in *Proc. IEEE Int. Conf. Communications ICC '09*, 2009, pp. 1–5.