On the Performance of Erasure Coding over Space DTNs*

Giorgos Papastergiou, Nikolaos Bezirgiannidis, and Vassilis Tsaoussidis

Space Internetworking Center, Department of Electrical and Computer Engineering,
Democritus University of Thrace, Xanthi, Greece
{gpapaste,nbezirgi,vassilis}@spice-center.org

Abstract. Erasure coding has attracted the attention of space research community due to its potential to present an alternative or complementary solution to ARQ schemes. Typically, erasure coding can enhance reliability and decrease delivery latency when long delays render ARQ-based solutions inefficient. In this paper, we explore the benefits of erasure coding for file transfers over space Delay Tolerant Networks, using a generic end-to-end mechanism built on top of the Bundle Protocol that incorporates LDPC codes along with an ARQ scheme. The results reveal significant insights on the tradeoff among efficient bandwidth exploitation and delivery latency. We quantify the performance gains when optimal erasure coding is applied and investigate in what extent theoretically optimal performance is affected when suboptimal code rates are used. Beyond that, we highlight the ability of erasure coding to provide different QoS to applications, in terms of file delivery latency, by properly tuning the code rate.

Keywords: Delay tolerant networking, Erasure coding, Deep-space communications.

1 Introduction

In this paper, we investigate mechanisms to explore erasure coding techniques in space communications without (i) expending unnecessarily significant bandwidth for error correction overhead that cannot have return in application throughput and (ii) over-investing in Automatic Repeat reQuest (ARQ) techniques when long delays dominate communication performance. Hence, we focus here on mechanisms and experiments that may promote our knowledge on how to dynamically administer the tradeoff between bandwidth and delay in space, with a minimal risk. Given the inherent advantage and significant standardization progress of Delay Tolerant Networking (DTN) in space communications [1], we explore this tradeoff within the space DTN framework.

* *The research leading to these results has received funding from the European Community's Seventh Framework Programme ([FP7/2007-2013_FP7-REGPOT-2010-1, SP4 Capacities, Coordination and Support Actions) under grant agreement* ***n° 264226*** *(project title: Space Internetworking Center-SPICE).* This presentation reflects only the authors views and the Community is not liable for any use that may be made of the information contained therein.

Y. Koucheryavy et al. (Eds.): WWIC 2012, LNCS 7277, pp. 269–281, 2012.

Space communication channels are characterized by significant shadowing and fading events, which strongly reduce signal-to-noise ratio and thus introduce bit errors within link frames. In addition, adverse weather conditions can cause long fading events or even link disconnections. In such cases, typical data-link recovery fails, resulting in bursty frame losses, occasionally in the order of tens to thousands of lost frames. Link layer failures are reflected on the upper protocol layers as packet erasures, i.e., missing packets that need to be retransmitted; however, long propagation delays and disruptions render ARQ solutions inefficient, since retransmission latency degrades network performance and extends communication (and thus data delivery) time significantly.

Delay Tolerant Networking [2] incorporates two significant reliability enhancements, that is, *custody transfer* and *storage capability*; their combined impact allows for retransmissions with reduced delay, since the source shifts data and responsibility gradually to intermediate nodes, towards the destination. Hence, lost data will be retransmitted faster, from nodes closer to destination. Inherently, therefore, DTN appears as a natural solution to improve reliability in space. However, shifting data custody towards the destination and thus gradually reducing retransmission delay is not sufficient solution in its own right. Typical erasure coding techniques along with ARQ may also need to be adjusted. Erasure codes [3] have been introduced as complementary mechanisms to ARQ solutions, employing Forward Error Correction (FEC) strategies at higher layers with a clear-cut goal to reduce the number of retransmission rounds. Erasure coding imposes by definition a tradeoff between efficient bandwidth exploitation and faster delivery; however, the question "*when and to what extent does extra and redundant transmission effort translate into better application throughput"* has not yet been adequately addressed. This issue becomes vital in space, where a false strategy to exploit the tradeoff may cause minutes or hours of waiting.

Given that erasure coding is a packet-level FEC technique, it can be incorporated within any protocol of the CCSDS DTN protocol stack that deals with data units (i.e., packets, frames, etc.) and since the DTN architecture is accomplished in a hop-by-hop manner, these solutions will essentially follow the same approach. Although hop-by-hop erasure coding can be tuned for the separate link conditions, it poses several challenges in terms of interoperability and cross-support. On the contrary, the deployment of erasure coding in an end-to-end "transport layer" architecturally placed above the DTN architecture could offer several advantages. From a technical point of view, an end-to-end approach moves complexity towards the ends of the communication system and leads to lower total processing delays, since encoding/decoding processes reside only at the end nodes. From a networking point of view, implementing erasure coding in an upper layer allows for handling packet erasures that are not related to frame losses only, but span across the whole protocol stack (e.g., packet erasures caused by storage congestion or erroneous route calculations at the bundle layer). Furthermore, this approach allows applications to identify certain QoS requirements to the "transport" service below (e.g., delay constraints, packet sizes, etc.); these requirements are associated to the complete end-to-end path and code rate is adapted accordingly.

We note that a performance comparison between hop-by-hop and end-to-end approaches is out of the scope of this paper. Here, we evaluate the performance gains of erasure coding against typical ARQ solutions and investigate the dynamics of the associated trade-offs. Although our contribution to exploiting these dynamics includes a novel generic end-to-end erasure coding protocol, which is placed architecturally between the application and the Bundle Protocol [4], in this initial study a single hop topology is considered. This experimental protocol incorporates erasure coding operating on a per-packet basis, based on block Low Density Parity Check (LDPC) codes [5], and also packet-oriented retransmission of encoding packets whenever decoding is unsuccessful. The proposed coding strategy differs from typical Type-II Hybrid ARQ strategies in that the lost encoding packets are retransmitted and no additional recovery packets are generated. Real-time experiments are conducted using our DTN Testbed [6, 7] and file transfers of different sizes are considered.

The conclusions presented in this work quantify but also qualify this tradeoff in the context of:

a) the optimal gain of erasure coding for file transferring
b) the impact of over- / under-estimation of channel packet erasure rate (PER) on file delivery time and the associated waste of bandwidth resources
c) the capability of an end-to-end "transport layer" erasure coding service to administer QoS in terms of delivery latency

The remainder of the paper is organized as follows: In Section 2 we discuss the related work and we highlight our perspective within this context. In Section 3 we briefly describe the proposed erasure coding experimental protocol. In Section 4 we elaborate on the experimental methodology, metrics and evaluation cases. We present the results of our experimental analysis in Section 5, and finally, in Section 6 we conclude the paper and provide some directions for future research.

2 Related Work

Appropriate positioning of erasure codes within the CCSDS protocol stack is an issue that triggered an interesting debate [8] in the CCSDS community. For example, erasure codes may be implemented as application layer solutions [9], as mechanisms of CFDP [10, 11] or at the DTN Bundle Protocol extensions [12]. Furthermore, authors in [12] compare two alternative approaches to support packet-level FEC: CFDP and DTN bundle protocol extensions. However, they do not conclude in favor of one or another.

Other approaches to incorporate erasure coding in space communications that however do not follow CCSDS architecture also exist. RCP-Planet [13] is an end-to-end rate control protocol for Interplanetary Internet (IPN) that targets the delivery of real-time application data. The protocol incorporates a probing rate control scheme to cope with link congestion and error rate, in conjunction with a packet-level FEC that is based on Tornado codes. The term "real-time" is rather a euphemism for channels with high propagation delays. However it reflects the time constraints of space applications. In [14] the authors propose Uni-DTN, a non-CCSDS DTN convergence layer protocol for unidirectional transport to provide scalability for both unicast and

multicast distribution of DTN bundles. Although some ideas included in [14] are promising, they do not target space environments and therefore, comparisons cannot be made easily.

Although simulation results across different space environments (either near-Earth/Cislunar [10] or deep-space [11]) are available for some of the aforementioned approaches, they are mainly scenario-oriented and do not highlight the specific tradeoffs of erasure coding versus ARQ schemes. For example, evaluation results presented in [10, 11] are strictly confined within specific and predetermined QoS parameters (delivery time / loss probability) and classes of space data traffic and hence, conclusions can only be confined within this specific context.

On the contrary, we attempt to go beyond the scenario-confined conclusions and investigate the tradeoff *per se*; that is to focus on the benefits of erasure coding along with ARQ schemes for future space internetworking architecture, in a way that scenario-independent conclusions about its performance can be drawn. In order to be able to generalize our conclusions, we apply a generic ARQ scheme that incorporates LDPC codes.

In this work, we consider deep-space communication links, where the performance of typical ARQ solutions is highly affected. Our results reveal interesting dynamics that can constitute the basis for further investigations and guide the design of efficient protocol solutions incorporating erasure codes in the future. To the best of our knowledge, this is the first attempt to evaluate the performance of erasure coding using a real DTN testbed that fully implements the standard DTN architecture.

3 Erasure Coding Experimental Protocol

In this section, we describe the erasure coding (EC) mechanism used in our experiments. EC is incorporated in an end-to-end "transport layer" protocol built on top of the Bundle Protocol and operates only at the endpoints of the communication system. Then, EC is integrated into Interplanetary Overlay Network (ION) DTN implementation [15]. EC mechanism is based on two, large-block, patent-free LDPC codes, namely LDPC Staircase and LDPC Triangle, which are specified in [16]. LDPC Staircase is used in all experiments, since this code presents lower inefficiency ratios [17] for the code rates we are experimenting with. Since only file transfers are considered, complete files are passed by the application to EC for transmission.

Each time a file transmission is requested, the file is handed down by the application to EC for transmission. The file is partitioned into N number of LDPC blocks, where $N = fileSize / maximumBlockLength$. *MaximumBlockLength* is a configuration parameter defined *a priori.* In our experiments, each file typically constitutes a single LDPC block. However, experiments have been conducted also with a 60Mbyte file partitioned into four LDPC blocks of 15MByte length each, in order to specifically investigate the impact of block size itself on EC performance.

Each LDPC block is segmented into k fixed-length source packets. The k source packets along with the value of the code rate are passed on to the LDPC encoder and, consequently, n encoding packets are created, where $n = k / code\_rate$.

Since both LDPC Triangle and LDPC Staircase are systematic codes, the first k encoding packets are the k source packets. The remaining n-k encoding packets are FEC packets. Ideally, code rates could statistically exploit the historical characteristics of

the channel. Assuming that channel PER is known *a priori*, optimal code rate (i.e., the code rate that suffices to protect a file transmission, given some PER) is defined as follows:

$$R = \frac{1 - PER}{1 + \varepsilon \cdot (1 - PER)} \quad (1)$$

where ε is the LDPC inefficiency. In order for decoding to be successful, $(1+\varepsilon)k$ encoding packets are required at the receiver. We note that channel PER accounts for total packet erasures that occur across the end-to-end path.

All encoding packets are passed to BP *in order*, requesting unreliable transmission and each encoding packet is encapsulated into a single bundle. The reception of the last encoding packet (*checkpoint* packet) always triggers the transmission of an *acknowledgement* indicating either the successful (*Ack*) or unsuccessful (*Selective Negative Acknowledgment*, *SNAck*) block reception; hence, *acknowledgment* delivery must be guaranteed. For this reason, a retransmission timer for the last packet of each encoded LDPC block is set. Upon the arrival of an *Ack*, the timer is cancelled; otherwise, upon expiration of this timer, the packet is retransmitted. Upon the reception of a *SNAck*, lost encoding packets are retransmitted. *SNAcks* inform the sender about missing encoding packets, in the same way as in [18]. In order to guarantee reliability, the last packet of each retransmission round also triggers the reliable transmission of *acknowledgments*. When decoding succeeds, LDPC blocks are aggregated into a single application data unit and delivered to the application.

4 Experimental Methodology

4.1 Scenario - Parameters

Our research purpose is to evaluate the potential of erasure coding in deep-space environments. In order to emulate long delays, high error rates, and asymmetric space link channels, ESA DTN testbed [6, 7] established in Space Internetworking Center [19] was used. The testbed allows for emulating current and future DTN-based space communication scenarios and uses Network Emulator (Netem, [20]) to emulate space conditions (i.e., error rates, propagation delays and data rates). Our evaluation scenario is based on a deep-space mission paradigm, where an *in-situ* element that is used for planet exploration relays the observed scientific data for further processing towards Earth base stations. In this initial evaluation, we consider the deep-space link only, where the performance of typical ARQ-based solutions may degrade. In particular, we consider one-hop file transmissions from a Mars orbiter towards Earth. Communication parameters used were taken from Mars missions [21, 22].

Nevertheless, in order to reduce emulation time, conduct more experiment repetitions and increase statistical robustness, we choose to keep the *Bandwidth-Delay Product* (*BDP*) of the deep-space link fixed, by increasing bandwidth and decreasing propagation delay correspondingly. The results are normalized accordingly and thus independent conclusions can be drawn. For example, we show below that for fixed

BDP and PER, file delivery latency normalized based on the RTT depends on the BDP alone (Eq. 5).

Since the BDP remains stable, its capacity in terms of packets is the same. For given PER, the number of retransmitted packets (*rtxData*) and the number of retransmission rounds (*RtxRounds)* cannot pose statistical arguments. Assuming further that queuing and processing delays are negligible, total file delivery latency can be expressed as:

$$FDL_{total} = D_{pr} + D_{tr} \tag{2}$$

where D_{pr} is the total propagation delay, and D_{tr} the total transmission delay. In particular:

$$D_{tr} = \frac{(fileData + rtxData + fecData)}{BW} \tag{3}$$

$$D_{pr} = \frac{RTT}{2} + RTT \cdot RtxRounds = RTT \cdot (\frac{1}{2} + RtxRounds) \tag{4}$$

Thus, based on Equations 2-4 normalized file delivery latency (*NDL*) can be expressed as:

$$NDL = \frac{1}{2} + RtxRounds + \frac{(fileData + rtxData + fecData)}{BDP} \tag{5}$$

The downlink BDP used in our experiments was 120 Megabits (i.e., 15 Mbytes), which is the product of 20 min *RTT* and 100 Kbps downlink data rate. The uplink BDP used is 1.2 Megabits, hence providing the deep-space channel asymmetry. In order to investigate the tradeoffs of erasure coding in both a scenario- and link-independent way, all file sizes used in our experiments are expressed in correspondence to the downlink BDP. In particular, file sizes vary from 0.5xBDP (i.e., 7.5 MBytes) to 4xBDP (i.e., 60 MBytes). Files were truncated into bundles with payload size equal to 1024 Bytes.

We consider two distinct evaluation cases: In *Case 1* we evaluate the performance gains of the EC mechanism and compare it with a simple ARQ-based scheme. This scheme shares the same mechanisms for file partitioning, block segmentation, and packets transmission and retransmission with the EC mechanism. The only difference between these two schemes is that in the former case no encoding is performed and thus only the *k* source packets are transmitted. In case no file partitioning into blocks is performed, its operation is equivalent to the CFDP deferred NAK mode [23]. Typical PERs for deep-space conditions are considered. Additionally, higher PERs are used to emulate extreme space weather conditions (e.g., solar winds). Thus, PER varies between 0% and 30%. Code rate is adjusted according to Eq. 1. In *Case 2*, we evaluate how deviations in channel PER estimation affect the performance gains of the EC mechanism; in parallel we investigate in what extent end-to-end erasure coding can differentiate the service provided to file transfer applications. In this case,

PER remains fixed at 20%, while PER estimations vary between 0% and 35%. Code rate is adjusted based on the estimated PER according to Eq. 1, where inefficiency ratio ε is optimized for each PER and its value varies from $\varepsilon = 0.03$ for PER = 5% to $\varepsilon = 0.09$, for PER = 35%.

A simple independent packet erasure model is considered, in which a packet is lost with a probability equal to PER. A summary of the scenario parameters is given in Table 1 below. Each experiment was repeated adequate times and both average values and 95% confidence intervals are presented.

Table 1. Scenario Parameters and Values

Parameter	Value
EC/ARQ packet size (Kbytes)	1024
File Size (Mbytes)	7.5, 15, 30, 60
Downlink BDP (Mbits)	120
Uplink BDP (Mbits)	1.2
Propagation Delay (sec)	40
Uplink Rate (kbps)	15
Downlink Rate (kbps)	1500
PER (%)	0, 5, 10, 15, 20, 25, 30, 35

4.2 Performance Metrics

In order to examine the tradeoff between the gain in data delivery latency and the waste of bandwidth imposed by redundant transmissions, we evaluate the performance of erasure coding against different performance metrics. Delivery latency is represented by two metrics: *Normalized Delivery Latency* (*NDL*) and *Normalized Delivery Latency Gain (NDLG). NDL* is the file delivery latency normalized based on the RTT. *NDLG* is the gain percentage-wise in file delivery time when the EC mechanism is applied, with respect to the simple ARQ-based mechanism.

Data redundancy imposed either by retransmitted packets or by FEC packets is evaluated using two metrics, *Normalized Redundancy (NR)* and *Normalized Redundancy Loss* (*NRL*). *NR* is the total number of redundant bytes normalized based on the BDP. When the simple ARQ-based mechanism is applied, *NR* accounts only for the retransmitted packets, while in the cases where the EC mechanism is applied, *NR* considers also the *n-k* redundant packets initially transmitted. *NRL* is the increase percentage-wise in *NR* when the EC mechanism is applied, with respect to the simple ARQ-based mechanism.

5 Experimental Results

Figures 1-3 present the experimental results for the first evaluation case, as described in Section 4.1. Since code rates used for the EC mechanism are optimal for each

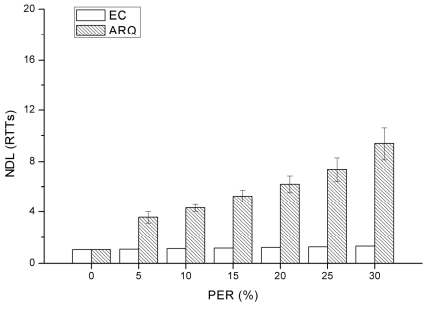
corresponding PER value, EC mechanism achieves in all cases successful file decoding and delivery after the first transmission round, thus avoiding retransmissions. In Fig. 1 we observe that, as PER increases, *NDL* for both schemes increases as well. This increase, however, is remarkably more significant when the ARQ-based mechanism is applied; the higher the PER, the higher the number of retransmission rounds required. On the contrary, when EC mechanism is applied, *NDL* is affected only by the additional delay required for the transmission of the encoding packets; the higher the PER, the higher the number n of the encoding packets (see Eq. 1).

As far as *NDL* is concerned, two major conclusions can be drawn. Firstly, we observe that as PER increases, the benefits of erasure coding, in terms of *NDL,* become more significant. Indeed, as shown in Figure 3, *NDLG* increases considerably as PER increases. The maximum observed gain in *NDL* is approximately 86%, when file size is equal to 0.5xBDP and PER is 30%. We further observe that the gain in *NDL* is significantly affected by the file size. For a given PER, as file size increases, *NDLG* decreases, respectively. For example, when PER is 30% and for file sizes equal to 1xBDP and 4xBDP, *NDLG* decreases from 0.82 to 0.6, respectively. This is explained by the fact that as file size increases, transmission latency comprises a significant portion of the total delivery time.

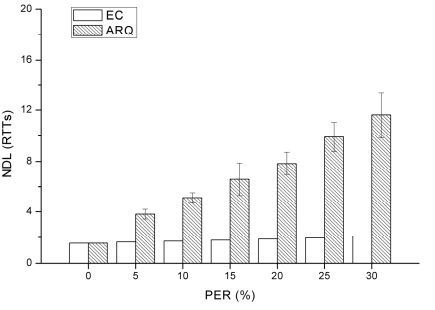
Even though EC reduces file delivery latency significantly, it necessarily imposes a redundancy overhead due to LDPC decoding inefficiency. In other words, although code rate is adjusted properly to protect file transmission for each given PER value, compared to Maximum Distance Separable (MDS) codes, LDPC codes require additional encoding packets for successful decoding. Data redundancy for both schemes and for different PER values is illustrated in Figures 2 and 3. *NR* for both mechanisms presents similar behavior for all file sizes and increases proportionally with file size. Therefore, we omit from Fig. 2 the corresponding graphs for the 1xBDP and 4xBDP file sizes. As expected, *NR* is significantly higher when EC mechanism is applied.

Although *NR* depends on file size, *NRL* does not; it depends only on PER and therefore it is depicted in Fig. 3 with a single line. As shown in Fig. 3, as PER increases, *NRL* decreases. That is, as PER increases, the contribution of LDPC inefficiency in total *NR* becomes, percentage-wise, less significant. For example, *NRL* is almost 0.6 when PER is 5% and decreases to around 0.27 for PER = 30%.

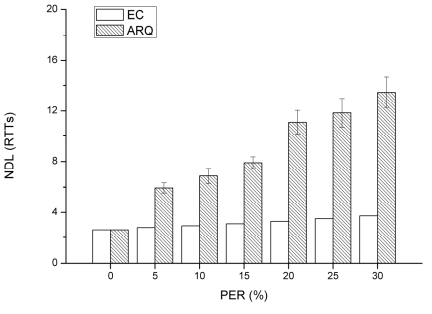
Fig. 3 gives a detailed insight about the tradeoff between the gain in file delivery latency and the waste of bandwidth due to the increased redundancy overhead. We observe that for low PERs (e.g., 5%) the gain in *NDL*, when erasure coding is applied, is comparable to the waste of bandwidth, percentage-wise. However, as PER increases, we observe that *NDLG* increases, while at the same time *NRL* decreases. Fig. 3 can constitute the basis for constructing a cost-based graph, where different cost weights can apply to each performance metric; the intersecting point between the two cost-weighted lines can indicate a performance threshold after which erasure coding, compared to typical ARQ-based solutions, is beneficial.



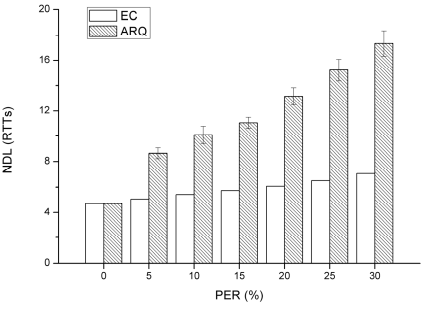
(a) File Size = 0.5 BDP



(b) File Size = 1 BDP

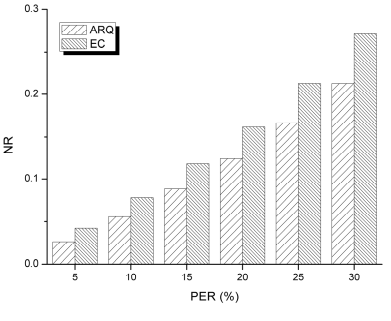


(c) File Size = 2 BDP

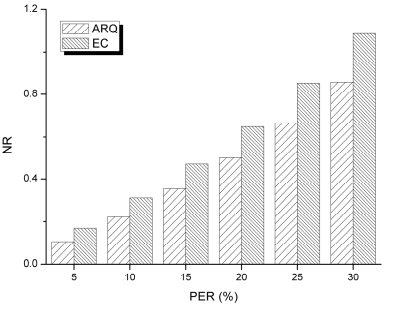


(d) File Size = 4 BDP

Fig. 1. Case 1 – Normalized Delivery Latency



(a) File Size = 0.5 BDP



(b) File size = 2 BDP

Fig. 2. Case 1 – Normalized Redundancy

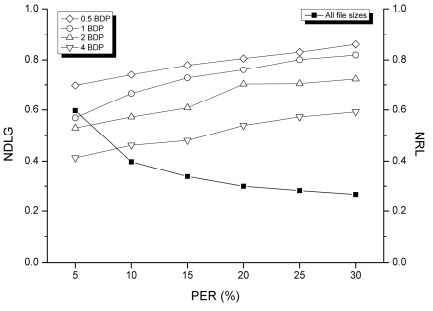
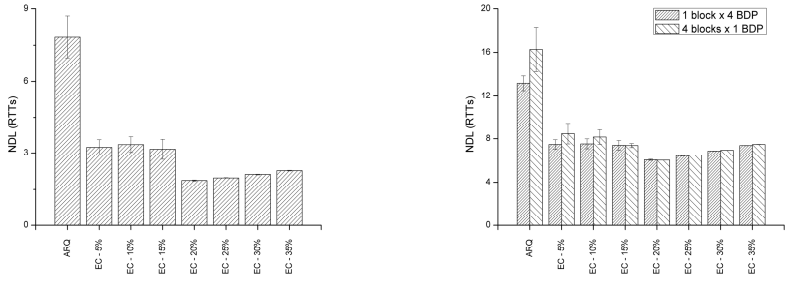


Fig. 3. Case 1 - NDLG vs NRL

In *Case 1* we have assumed that PER is known *a priori* and that code rate was adjusted accordingly (Eq. 1). This is, however, an ideal case and such knowledge is practically nonexistent. Thus, in the second evaluation case (*Case 2*), we investigate performance tradeoffs when predicted PER deviates from actual PER. Figures 4-6 present the evaluation results for *Case 2*.

In Fig. 4, *NDL* for different predicted PER values is shown and compared with the simple ARQ scheme. Two different file sizes are considered: 1xBDP and 4xBDP. In order to investigate in what extent file partitioning into blocks affects performance, we also consider the case where the 4xBDP file is partitioned into four equal-sized blocks.



(a) File size = 1 BDP

b) File size = 4 BDP

Fig. 4. Case 2 – Normalized Delivery Latency for different PER predictions. Actual PER = 20%.

As observed in Fig. 4, even when the predicted PER is lower than the actual, *NDL* decreases significantly when EC mechanism is applied. An interesting remark is that regardless of how low the predicted PER is, the reduction in *NDL* remains almost the same. As expected, the lowest *NDL* is obtained when the predicted PER matches the actual (20%). For higher predicted PERs, there is an increase in *NDL*. This is explained by the fact that when the code rate is lower than the theoretically optimal

(i.e., it targets on protecting higher PERs), decoding inefficiency increases and this affects *NDL*. The impact of such overestimation becomes more significant as file size increases. For example, when file size is 4xBDP and predicted PER is 35%, *NDLG* is comparable to the *NDLG* observed for PERs 5-15% (Fig. 6).

From a different point of view, assuming that actual PER is known *a priori*, we observe that erasure coding, when combined with ARQ-based schemes, can be used as a means to provide coarse-grained service differentiation among file transfers, without requiring any modifications in the underlying network. In particular, considering predicted PER as a configuration parameter, we observe in Fig. 4 that three different classes of service can be provided, in terms of delivery latency, by configuring predicted PER to 0%, 5% and 20%, respectively. Adjusting predicted PER to higher values (e.g., 10-15% and 25%-35%) results in similar *NDL* for the latter two classes, but it wastes bandwidth resources unnecessarily.

Furthermore, we note that the gain in *NDL* is significantly affected by the file size and, as already observed in *Case 1*, *NDLG* decreases when file size increases (Fig. 6). Regarding the multiple-block file transmission, results show that in this case the performance of the simple ARQ-based scheme is considerably affected (Fig. 4b). In particular, when file is segmented into multiple blocks, *NDL* increases by 3 RTTs. This is due to the fact that when multiple blocks are used, the number of *checkpoint* packets increases as well. As a consequence, more *checkpoint* packets are statistically lost, requiring retransmission and thus extending delivery latency. On the contrary, *NDL* achieved by the EC mechanism appears to be less affected by multiple-block transmission and only for lower predicted PERs (5% and 10%). Thus, it becomes clear that when file is partitioned into multiple blocks, the gain in *NDL* with EC mechanism is even higher. Fig. 6 verifies this conclusion.

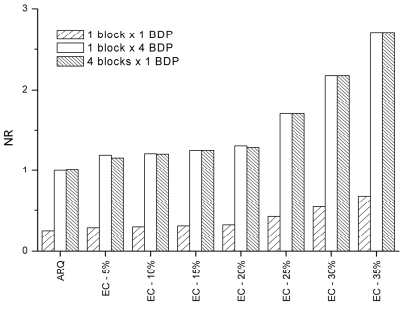


Fig. 5. Case 2 – Normalized Redundancy

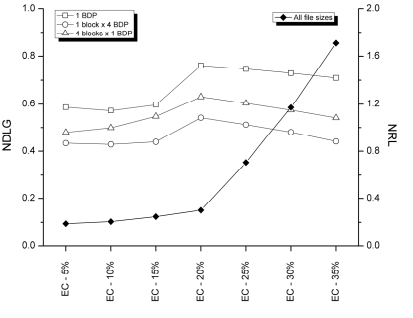


Fig. 6. Case 2 – NDLG vs NRL

Similarly to *Case 1, NR* presents the same behavior for all file sizes and increases proportionally to file size (Fig. 5). Besides, *NRL* is independent to the file size and depends only on the value of the predicted PER (Fig. 6). As shown in Fig. 5, *NR* subtly decreases when predicted PER gets lower than the actual (20%), and remains higher with respect to the simple ARQ-based mechanism. For higher values of predicted PER, however, *NR* is considerably affected. In particular, in Fig. 6 we see that when

PER prediction is 35%, *NRL* increases to 1.7, which can be interpreted as a 170% increase in redundancy, compared to the ARQ-based scheme. Multiple-block configuration imposes almost the same *NR* compared to single-block configuration. Finally, we note again that a cost-weighted graph based on Fig. 6 can be constructed in order to investigate the threshold after which erasure coding is beneficial, according to a cost function.

6 Conclusions – Future Work

In this work, we have examined the tradeoff between file delivery latency and redundancy overhead when erasure coding coupled with ARQ-based schemes is applied. An end-to-end transport protocol that was built on top of BP and which incorporates the EC mechanism introduced in this paper, was implemented and deployed into ESA DTN testbed, established in Space Internetworking Center. Results revealed significant insights on the performance tradeoffs imposed by erasure coding. Scenario-independent conclusions about when and to what extent erasure coding is beneficial in such environments were drawn. Additionally, results gave prominence to the capability of end-to-end erasure coding to administer QoS in terms of file delivery latency.

Although our proposed protocol solution comprises an end-to-end solution, in this initial work we have considered a single deep-space communication link. It is in our future intentions to broaden the analysis and investigate performance tradeoffs in multi-hop scenarios, where alternative paths may exist, communication links are characterized by varying propagation delays and PERs, and packet erasures are also caused by storage congestion. This analysis will further reveal the advantages and disadvantages of end-to-end approaches against hop-by-hop solutions. Finally, we intend to examine the applicability of adaptive erasure coding in space DTNs, where code rate is adjusted based on channel observations.

References

1. Rationale, Scenarios, and Requirements for DTN in Space. CCSDS Draft Green Book (March 2010)
2. Cerf, V., Burleigh, S., Hooke, A., Torgerson, L., Durst, R., Scott, K., Fall, K., Weiss, H.: Delay-Tolerant Networking Architecture. IETF RFC 4838, Informational (April 2007)
3. Rizzo, L.: Effective Erasure Codes for Reliable Computer Communication Protocols. ACM SIGCOMM Computer Communication Review 27, 24–36 (1997)
4. Scott, K., Burleigh, S.: Bundle Protocol Specification. IETF RFC 5050, experimental (November 2007)
5. Gallager, R.: Low Density Parity-Check Codes. MIT Press, Cambridge (1963)
6. Koutsogiannis, E., Diamantopoulos, S., Papastergiou, G., Komnios, I., Aggelis, A., Peccia, N.: Experiences from architecting a DTN Testbed. Journal of Internet Engineering 3(1) (2009)

7. Bezirgiannidis, N., Tsaoussidis, V.: Packet size and DTN transport service: Evaluation on a DTN Testbed. In: International Congress on Ultra Modern Telecommunications and Control Systems and Workshops, ICUMT, Moscow, pp. 1198–1205 (2010)
8. Cola, T.: Use of Erasure Codes in CCSDS Upper Layers: Motivation and Implementation. In: CCSDS Meeting, Portsmouth (May 2010)
9. Paolini, E., Varrella, M., Chiani, M., Calzolari, G.: Recovering from Packet Losses in CCSDS Links. In: IEEE Advanced Satellite Mobile Systems, ASMS, pp. 283–288 (2008)
10. Cola, T., Ernst, H., Marchese, M.: Performance analysis of CCSDS File Delivery Protocol and erasure coding techniques in deep space environments. Elsevier Computer Networks 51(14), 4032–4049 (2007)
11. Cola, T., Ernst, H., Marchese, M.: Application of Long Erasure Codes and ARQ Schemes for Achieving High Data Transfer Performance Over Long Delay Networks. Signals and Communication Technology 5, 643–656 (2008)
12. Cola, T.: A protocol design for incorporating erasure codes within CCSDS: The case of DTN protocol architecture. In: IEEE Advanced Satellite Multimedia Systems Conference, ASMA and The 11th Signal Processing for Space Communications Workshop, SPSC, pp. 68–73 (2010)
13. Fang, J., Akyildiz, I.: RCP-Planet: A Rate Control Protocol for InterPlaNetary Internet. International Journal of Satellite Communications and Networking 25(2), 167–194 (2007)
14. Kutscher, D., Loos, K., Greifenberg, J.: Uni-DTN: A DTN Convergence Layer Protocol for Unidirectional Transport. Work in progress as an internet-draft, draft-kutscher-dtnrg-uni-clayer-00 (2007)
15. Jet Propulsion Laboratory. Interplanetary Overlay Network, https://ion.ocp.ohiou.edu/
16. Roca, V., Neumann, C., Furodet, C.: Low Density Parity Check (LDPC) Staircase and Triangle Forward Error Correction (FEC) Schemes. IETF RMT Working Group, RFC 5170 (June 2008)
17. Roca, V., Neumann, C.: Design, Evaluation and Comparison of Four Large Block FEC Codecs, LDPC, LDGM, LDGM Staircase and LDGM Triangle, plus a Reed-Solomon Small Block FEC Codec. INRIA Research Report RR-5225 (June 2004)
18. Papastergiou, G., Psaras, I., Tsaoussidis, V.: Deep-Space Transport Protocol: A Novel Transport Scheme for Space DTNs. Computer Communications (COMCOM). Special Issue on Delay-/Disruption-Tolerant Networks 32(16), 1757–1767 (2009)
19. Space Internetworking Center, http://spice-center.org
20. Hemminger, S.: Network Emulation with NetEm. In: 6th Australia's National Linux Conference, LCA 2005, Canberra, Australia (April 2005)
21. European Space Agency, Mars Express, `http://www.esa.int/esaMI/Mars_Express/`
22. National Aeronautics Space Administration, Mars Exploration Program, `http://mars.jpl.nasa.gov/`
23. CCSDS File Delivery Protocol (CFDP). CCSDS Blue Book (January 2007)