

Rethinking Reliability for Long-Delay Networks

Jérôme Lacan
 Université de Toulouse
 DMIA-ISAE, LAAS-CNRS
 Toulouse, France
 jerome.lacan@isae.fr

Emmanuel Lochin
 Université de Toulouse
 DMIA-ISAE, LAAS-CNRS
 Toulouse, France
 emmanuel.lochin@isae.fr

Abstract—Delay Tolerant Networking (DTN) is currently an open research area following the interest of space companies in the deployment of Internet protocols for the space Internet. Thus, these last years have seen an increase in the number of DTN protocol proposals such as Saratoga or LTP-T. However, the goal of these protocols are more to send much error-free data during a short contact time rather than operating to a strictly speaking reliable data transfer. Beside this, several research work have proposed efficient acknowledgment schemes based on the SNACK mechanism. However, these acknowledgement strategies are not compliant with the DTN protocol principle. In this paper, we propose a novel reliability mechanism with an implicit acknowledgment strategy that could be used either within these new DTN proposals or in the context of multicast transport protocols. This proposal is based on a new erasure coding concept specifically designed to operate efficient reliable transfer over bi-directional links.

I. INTRODUCTION

In the today Internet, TCP enables by default the Selective ACKnowledgement (SACK) [1] mechanism to perform reliability and the TCP/SNACK variant [2] is preferably used over satellite links. In the context of DTN, the store and forward nature and the potential disruption events inherent to this architecture might prevent the use of TCP. Indeed, as emphasized in [3], TCP is useful for end-to-end communications across a shared Internet where fairness remains the key parameter and not for store-and-forward communications across a single unshared link where TCP is known to obtain disappointing performances. However, shared satellite links are also used in the context of broadband satellite systems (such as IPSTAR [4] or WildBlue [5]) and reliability is always needed in the context of bulk data transfer.

Obviously, the use of a reliability mechanism based on retransmission such as SNACK can strongly be counterproductive in a DTN context. This is one of the main reason that pushed DTN protocols' designer to see in the Forward Error Coding (FEC) mechanism the best solution to improve reliability [3]. Following this point, we introduce in this paper a novel erasure coding concept, resistant to acknowledgement losses, in order to improve reliability. One of the goal of this erasure code concept is to increase the reliability of DTN protocols such as Saratoga [3] and LTP-T [6] proposals to enlarge their spectrum of use to any kind of long-delay networks such as shared satellite and high BER links.

II. ON THE ERASURE CODES

Most of erasure codes¹ used over packet erasure channels are block codes [7]. This means at the encoder side, a set of repair packets (R) is built from a given set of source data packets (called after source packets and noted P) and at the decoder side, these repair packets can only be used to recover source packets from their corresponding set. If too many packets (among the source and repair packets) are lost during the transmission, the recovery of the missing source packets is then not possible. On the opposite, if only few packets are lost, some of the repair packets become useless. A solution to this problem, known as Hybrid FEC-ARQ (or H-ARQ) mechanism [7], is to use receiver's feedback to send additional repair packets or to adjust the redundancy level of the FEC to the observed packet loss rate. However, in a DTN context, large RTT can lead to very long delays to recover efficiently a packet.

The proposal in [8] is to reduce the decoding delay by using non-binary convolutional-based codes. The principle behind is that each repair packets is generated from a sliding window in the set of source packets. However, this mechanism is specifically defined for real-time applications and cannot be directly applied in the DTN context. As an example, it does not provide full reliability and does not integrate the receivers' feedbacks.

Our proposal can be considered as a mix of these different solutions. The main idea is to build the repair packets from a source packets window (the coding window) which is updated with the receiver's feedbacks. This update is done in a way that any source packets sent is included in the coding window while the sender does not receive any acknowledgement. The method used by the sender to generate a repair packet is simply a linear combination over a binary or non-binary finite field of the data source packets belonging to the coding window. We allow the user the choice of the coefficients as a trade-off between the best performance (with non-binary coefficients) and the system constraints (the user might prefer the use of binary codes in an energy constrained environment). The receiver tries to perform the inverse linear combinations to recover the missing source packets from the received data and repair packets.

¹also called FEC.

III. PROPOSAL OVERVIEW

We denote P_n the n^{th} packet sent (with P_1 is corresponding to the first packet) and R_i^j a repair packet for all in sequence packets ranging from P_i to P_j . This redundancy ratio is defined as the proportion of repair packets among the total number of packets sent. This ratio can be defined either by the application (following quality of service requirements) or by a cross-layer mechanism (following network characteristics such as BER). Eventually, this number might be dynamically changed during the data transfer.

Let's start with an example: we suppose 4 packets sent and the fourth one be a repair packet noted: R_1^3 . This repair packet is computed as follows: $R_i^j = \sum_{k=i}^j \alpha_k^{(i,j)} \cdot P_k$ where the $\alpha_k^{(i,j)}$ belong to a finite field fixed. Then, assuming the sender emits packets: (P_1, P_2, P_3, R_1^3) ; the repair packet R_1^3 allows to rebuild a lost packet among three.

Now, let's assume the sender transmits the five packets sequence: $(P_1, P_2, R_1^2, P_3, R_1^3)$. If (P_2, P_3) are lost, the remaining packets allow to rebuild the whole sequence. If (P_1, P_2) are lost, P_3 can be first "subtracted" from the repair packet R_1^3 by computing $R'_1 = R_1^3 - \alpha_3^{(1,3)} \cdot P_3$. We then have $(R_1^2, R_1^3)^T = G \cdot (P_1, P_2)$ where G is the following 2×2 -matrix:

$$G = \begin{pmatrix} \alpha_1^{(1,2)} & \alpha_2^{(1,2)} \\ \alpha_1^{(1,3)} & \alpha_2^{(1,3)} \end{pmatrix} \quad (1)$$

Clearly, rebuilding (P_1, P_2) from (R_1^2, R_1^3) can be done if and only if G is invertible since if G^{-1} exists, we have $(P_1, P_2) = G^{-1} \cdot (R_1^2, R_1^3)$. By using classical results on erasure codes, it can be shown that G has an extremely high probability of being invertible if the finite field is chosen sufficiently large².

This property leads to efficient transmission and lighten acknowledgement strategies. A broader example is given in Fig. 1. This figure shows the two windows handled by the receiver, one for the repair packets and one for the received packets.

In this example, during the data exchange, packet P_2 is lost. However, the repair packet R_1^2 successfully arrived allows to rebuild P_2 . Then, the receiver sends an acknowledgement packet (dashed line) to inform the receiver that it can compute the next redundancy packets from packet P_3 . Then, this acknowledgement is lost; however, this lost does not impact on the future of the transmission as the sender still computes a redundancy packet from P_1 . After this, we can see that P_3, P_4 and R_1^4 packets are lost. None of these packets need to be retransmitted as they are rebuilt thanks to the packets received from P_5 to R_1^8 . Indeed, as for matrix (1), we can also rebuild P_3, P_4 by firstly "subtracting" all the received source packets from the repair packets in order to obtain (R_1^6, R_1^8) as follows:

²Note that in this particular case, it is possible to choose these factors to ensure this inversion success.

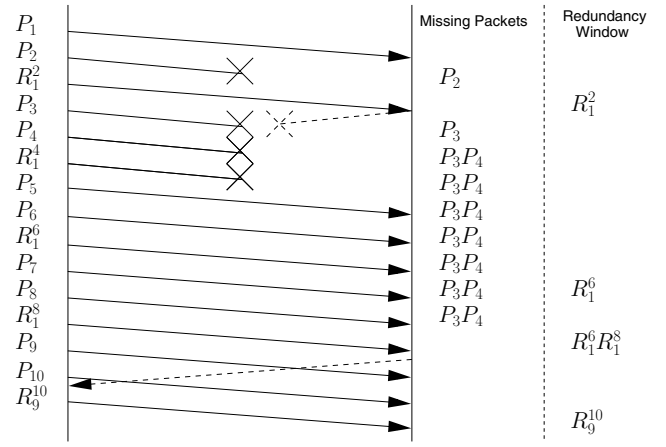


Figure 1. Reliability mechanism illustration

$$(R_1^6, R_1^8)^T = \begin{pmatrix} \alpha_3^{(1,6)} & \alpha_4^{(1,6)} \\ \alpha_1^{(1,8)} & \alpha_4^{(1,8)} \end{pmatrix} (P_3, P_4) \quad (2)$$

Then, the recovery of P_3, P_4 can be done by computing the inverse of the 2×2 -square matrix and by multiplying it by (R_1^6, R_1^8) .

Finally, the number of packets that can handle a repair packet R_i^j depends on the size of the finite field chosen. The larger is this number, the more is the computation complexity. In the case where the use of an acknowledgement path is possible, the dashed acknowledgement line allows to reset this counter as illustrated in Fig. 1. Indeed, this acknowledgement packet function is twofolds: first, it allows to reset the repair counter in order to decrease the computation time (in Fig. 1, after receiving the acknowledgement packet, the repair packet R_9^{10} starts from 9 and not from 1) and second, this confirms the sender that the packets sequence ranging from P_1 to P_8 are well-received. So, we can note that our mechanism does not depend on the acknowledgements received and is resistant to acknowledgements lost. Indeed, without any feedback received, the sender would send R_1^{10} without any impact on the communication reliability.

IV. PRELIMINARY MEASUREMENTS

In this section, we illustrate the behavior of this protocol by simulating a bulk data transfer over a lossy link without acknowledgement path. Then, as the packets lost recovery is strongly linked to the success of the matrix inversion process previously explained, we quantify the feasibility of this principle by presenting some matrix inversion statistics obtained by simulation.

We have implemented, as a traffic generator, our proposal and used the Linux Network Emulator (Netem) to implement the network losses and delay. We use the same redundancy ratio scheme (equivalent to 33%) as in section III (*i.e.* one redundancy packet sent each two source packets sent). The percentage of losses used during the experiment is ranging from 10% to 33% and the one way delay is set to 50ms

	Our proposal	FEC	Retransmission
Enable full reliability	YES	NO	YES
Improve transmission quality	YES	YES	NO
Real time transmission	YES	YES	NO
Delay tolerant	YES	YES	NO
ACK losses tolerant	YES	N/A	NO

Table I
COMPARISON BETWEEN SCHEMES

(equivalent to an initial RTT of 100ms). The packets are randomly dropped thus, no large burst occurs³. All packets have a fixed size of 1040 bytes and each next sent packet is spaced by one millisecond. The total number of packets (*i.e.* data and redundancy packets) sent during each experiment is 3000.

In order to assess the benefit of our proposal, we propose to compare our mechanism with a similar FEC encoding process. For this, we choose a FEC algorithm which create one redundancy packet every two packets sent.

We have to emphasize the difficulty of meaning when comparing our proposal with a FEC scheme or with a retransmission scheme (*e.g.* ARQ) as the goals between all these mechanisms strongly diverge. Indeed, our proposal attempts to implement a reliability mechanism without using packets retransmission through an erasure code which remains non-sensitive to the losses that might occur on the acknowledgement path. On the contrary, the goal of a FEC encoding mechanism is mainly to improve the link transmission quality while a retransmission scheme strictly focus on the success of the data transfer instead of the link occupancy or characteristics. In a DTN context, the DTN protocols recently proposed raise strong hypothesis on the link state in order the transfer succeed as a feedback path is not always available. Then, in this case, our proposal allows these protocols to use a mechanism allowing them to guarantee a kind of reliability of the data transfer. In order to clearly distinguish the goals of each schemes and point out the characteristics of our proposal, we list table I the characteristics of these different mechanisms.

Table II provides the results obtained by our simulation experiments. This table returns the data packets and redundancy packets lost during the experiments with a lossy channel of 10%, 20%, 30% and 33% packet loss rate. The *total packets lost* line allows us to verify the effective percentage of lost packets. As an example, we can see for the 33% experiment that the real ratio of dropped packets is slightly above this value.

Compared to the FEC scheme, we can see that our proposal

³Another measurements campaign is expected to study the case of bursty channels.

rebuild all lost packets when we are strictly below 33% of losses. With the 30% experiment, the 3.15% unbuilt packets are corresponding to the end of the transfer which unfortunately, did not provide enough redundancy packets to rebuild the whole file transferred. In Fig. 2, we represent the number of packets to rebuild with the number of redundancy packets available. When the latter is higher or equal to the former, the algorithm is able to rebuild all missing packets. In order to explain why these 3.15% of not rebuilt packets are not a problem, we have to closely look at this Fig. 2.

In Fig. 2(a), we can see that the algorithm can solve a square matrix up to 3x3 elements to rebuild 3 missing packets during the 10% experiment. During the 20% experiment, the maximum matrix size has reached 18x18 elements. Finally, the 3.15% missing packets to rebuild are represented Fig. 2(c) where at the end of transfer, the number of packets to rebuild was higher than the number of redundancy packets. This issue is more a protocol engineering problem rather than an erasure coding problem. Indeed, we can see in this first three experiments that our proposal strongly ensures a full data reliability during the transfer. Obviously, we cannot predict when the matrix will be built at the receiver side, however, we could estimate a peak matrix size. Keeping in mind that we did not use the acknowledgement path during our experiments, a possible solution to this problem is to transmit only redundancy packets at the end of the file transfer process and to wait for a final closing feedback packet from the receiver. Thus, this largest matrix estimation would allow the sender to optimally send specific redundancy packets in order to ensure the end transfer decoding. We reserve the estimation of the peak matrix size to a future contribution.

Finally, we show that our proposal does not outperform the FEC mechanism when the packet loss rate is higher than the redundancy ratio (33% in table II). Furthermore, Fig. 2(d) emphasizes that both data and redundancy packets lines diverge. However, in another context, this problem might be lead to an inappropriate redundancy ratio and as explained at the beginning of section III, a dynamic reconfiguration of this ratio is thus possible through a feedback message from the receiver. This result is expected as we do not propose a mechanism to improve only the quality of the link but also the data transfer reliability (see table I). This result emphasizes the difference between both concepts and illustrate how our proposal fills the gap in the context of DTN protocols.

In the last Fig. 3, we present an evaluation of the matrix inversion probability. These statistics has been collected with simulations performed in similar conditions than the simulations presented in Fig. 2. Although the use of structured matrices can improve these results, it is well known that random matrices have high probability of being invertible provided the finite field is sufficiently large (see *e.g.* [9]). We thus choose to generate coefficients $\alpha_k^{(i,j)}$ of a repair packet with an uniform pseudo-random generator where, for each packet, the seed is fixed from the timestamp of the packet. With this method, the sender and the receiver can determine all the linear coefficients from the timestamp of the packet.

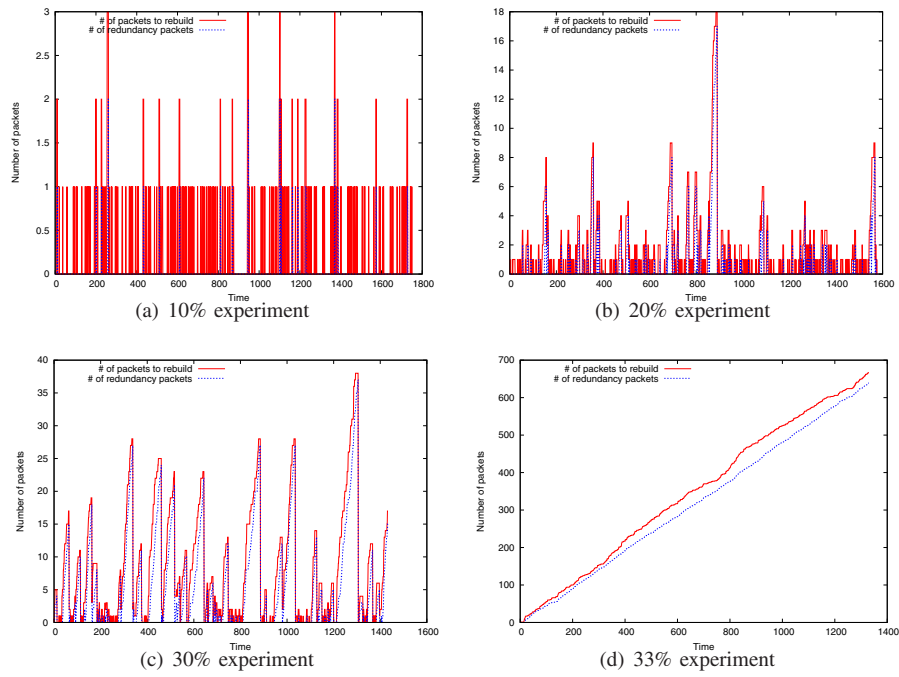


Figure 2. Number of packets to rebuild and number of redundancy packets received as a function of time during each experiments

PLR	10%	20%	30%	33%
PKTS STATS (%)				
Total pkts lost	10.51%	20.40%	28.80%	34.13%
Data pkts lost	12.42%	26.74%	37.51%	50.26%
Redundancy pkts lost	10.41%	23.45%	46.72%	55.03%
PKTS NOT REBUILT (%)				
Our proposal	0%	0%	3.15%	99.99%
FEC-like	12.84%	25.35%	37.36%	41.55%

Table II
EXPERIMENTS OVER A LOSSY PATH

In the presented simulations, we used the finite field \mathbb{F}_{256} , where the elements can be represented with bytes. The results of Fig. 3(b), which presents the respective percentage of successfully inverted matrices, confirm the theoretical results and prove that the matrices are invertible with an extremely large probability (greater than 99% in any case). This means that the recovery is possible as soon as the number of received repair packets reaches the number of lost data packets.

Fig. 3(a) gives the number of matrices to invert as a function of the matrices size. This parameter is interesting because the size of the matrix has consequences on:

- 1) the delay of recovery of the lost data packet and;
- 2) the amount of computations needed to perform the recovery.

We can observe that the size of the inverted matrix is very small (lower or equal to 2 in most cases) leading to short delay recovery and a low level of computations.

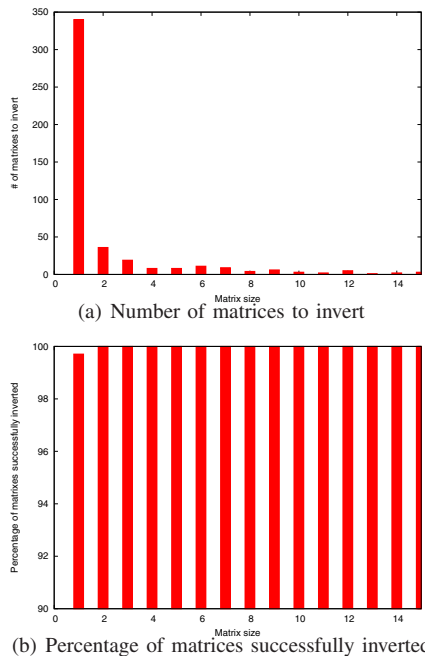


Figure 3. Statistics on matrices inversion for the 30% experiment

V. CONCLUSION

This paper has introduced a novel erasure code concept specifically designed for DTN communications. Preliminary

investigations have lead to interesting property of this code.
This solution:

- 1) allows a full reliability;
- 2) allows a fast recovery compared to block codes;
- 3) is not sensitive to acknowledgement losses;
- 4) avoids channel resources wasting due to non-useful retransmitted packets as the feedback scheme allows to optimize the number of redundancy packets sent to the most useful ones.

In a future work, we aim at demonstrating through analytical results the benefits of using our proposal. and that this proposal can also be used in the context of multicast communication. Furthermore, we argue this reliability scheme is not limited to the context of space Internet and could be use over a terrestrial Internet.

REFERENCES

- [1] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "TCP selective acknowledgment options," IETF, Request For Comments 2018, 1996.
- [2] R. Durst, G. Miller, and E. Travis, "TCP extensions for space communications," *the 2nd International Conference on Mobile Computing and Networking*, 1996.
- [3] L. Wood, W. Eddy, W. Ivancic, J. McKim, and C. Jackson, "Saratoga: a delay-tolerant networking convergence layer with efficient link utilization," in *IWSSC*, Salzburg, Austria, Sep. 2007.
- [4] "IPSTAR," <http://ipstar.com/>.
- [5] "Wildblue," <http://wildblue.com/>.
- [6] S. Farrell and V. Cahill, "Evaluating LTP-T: A DTN-Friendly transport protocol," in *IWSSC*.
- [7] S. Lin and D. Costello, *Error Control Coding: Fundamentals and Applications*. Prentice-Hall, Englewood Cliffs, NJ, 1983.
- [8] E. Martinian and C.-E. W. Sundberg, "Burst erasure correction codes with low decoding delay," *IEEE Transactions on Information Theory*, Oct. 2004.
- [9] J. Kahn and J. Komlós, "Singularity probabilities for random matrices over finite fields," *Combinatorics, Probability and Computing*, vol. 10, pp. 137 – 157, Oct. 2001.