Journal of Zhejiang University SCIENCE A
ISSN 1009-3095 (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Dependency-aware unequal erasure protection codes

BOUABDALLAH Amine[1,2,3], LACAN Jérôme[1,2,3]
([1]*Laboratory for Analysis and Architecture of Systems, Toulouse 31077, France*)
([2]*Applied Mathematics and Computer Science Department, ENSICA, Toulouse 31056, France*)
([3]*Telecommunications for Space and Aeronautic, Toulouse 31000, France*)
E-mail: {amine.bouabdallah; jerome.lacan}@ensica.fr

Received Dec. 5, 2005; revision accepted Feb. 25, 2006

Abstract: Classical unequal erasure protection schemes split data to be protected into classes which are encoded independently. The unequal protection scheme presented in this paper is based on an erasure code which encodes all the data together according to the existing dependencies. A simple algorithm generates dynamically the generator matrix of the erasure code according to the packets streams structure, i.e., the dependencies between the packets, and the rate of the code. This proposed erasure code was applied to a packetized MPEG4 stream transmitted over a packet erasure channel and compared with other classical protection schemes in terms of PSNR and MOS. It is shown that the proposed code allows keeping a high video quality-level in a larger packet loss rate range than the other protection schemes.

Key words: Data dependencies integration, Unequal erasure protection (UEP), Lossy networks, Reliable video transmissions, MPEG4 video codec
doi:10.1631/jzus.2006.AS0027 **Document code:** A **CLC number:** TN919.8

INTRODUCTION

The reliability is one of the major concerns for video transmissions over packet networks. Indeed, even a small packet loss rate could drastically decrease the quality level of the video.

The most classical solutions for general packet transmissions are based on the retransmissions of the lost packets. This solution, which is the simplest, is the most widely deployed for example by TCP. However, the main condition to deploy this solution is to have a return link that allows the receiver to inform the sender of lost packets through acknowledgments. The use of feedbacks is not well-adapted to all kinds of transmissions. For example, for multicast transmissions with large groups of receivers, a large number of simultaneous acknowledgments can produce congestions near the sender (feedback implosion). Another drawback of the feedback use is the increase of the latency of transmissions which cannot be supported by real-time applications. It is particularly the case for packet video transmissions.

When no return channel is available or when feedbacks are not adapted, proactive approaches can be used. These approaches are based on erasure codes (Rizzo, 1997) that allow the construction of redundant packets which are used by the receiver(s) to recover the lost packets. From now, for multicast and/or multimedia transmissions, these solutions are undoubtedly recognized. The recent work of several normalization organisms (IEFT: RMT and AVT groups; 3gpp: MBMS group; ETSI: DVB-H application layer FEC) on this subject testify to the validity of this solutions.

On top of proactively protecting the transmission, erasure codes schemes can be adapted to the data properties by allocating more protection to specific parts of the data. This is especially interesting for multimedia data which can be of unequal importance for the decoder.

Several works addressed this issue, the most classical is the Priority Encoding Transmission (PET) system which allows the sender to decompose the data into classes of given importance and which ensures that some full classes are received provided a determined amount of data is received (Albanese *et al.*, 1996). This system was extensively studied in numerous papers such e.g. (Mohr *et al.*, 2000; Chou *et al.*, 2003). The different classes of data have a global dependency relation and the system ensures (by construction) that if a class is received, all the classes with a higher level of importance are also received. This is the basic property of all the PET-based solutions such as e.g. (Liebl *et al.*, 2004).

From our point of view, one drawback of this solution is that the dependencies between the data is integrated at class-level and not at data-level as it could be done in a sequence of video frames. The aim of the work proposed here is to define a system of data protection integrating the dependencies at the packet-level by keeping the data dependency graph produced by the source encoder.

This system is described through the construction of a generator matrix of an erasure code. More specifically, we explain how to integrate the data dependencies in this generator matrix by introducing some simple and intuitive rules. An example of these rules is that if a redundant packet protects a packet belonging to a given video frame, then it must protect all the packets of this frame and all the packets belonging to the parent frame of the considered frame. This coding and the decoding algorithms are based on classical erasure code coding and decoding algorithms.

After Section 2 devoted to erasure codes, the construction is presented in Section 3. A description of the implementation of the proposed scheme and the simulation results are presented in Section 4. We conclude and present future work in Section 5.

ERASURE CODES

The principle of an erasure code is to add redundancy in order to cope with possible localized losses of data. The data units are correctly received or lost on the packet networks. The channel is then a Packet Erasure Channel. The erasure codes are usually derived from error correcting codes used on the bit or byte-error channel. They are defined over a finite field $\boldsymbol{F}$ that is defined as a finite set of p^m elements (where p is a prime number) which have a field structure.

An erasure code is usually defined as a linear vector space over $\boldsymbol{F}$. It can then be defined by a generator matrix $\boldsymbol{G}$ (a basis of the vector space) which is used for the encoding. Indeed, for an erasure code protecting k packets with $n-k$ additional redundant packets, $\boldsymbol{G}$ is a $k\times n$-matrix and the encoding of a vector $\boldsymbol{v}$ of k packets is done by performing the vector-matrix multiplication $\boldsymbol{v}\times\boldsymbol{G}$. For the decoding, the principle consists of considering the submatrix of the generator matrix corresponding to the received packets, then to invert this matrix (if it is possible) and to multiply this inverse by the received packets.

Several families of erasure codes are used. Maximum-Distance Separable (MDS) codes (Rizzo, 1997) are optimal in terms of erasure correction capability per bloc. Indeed, they are able to recover k information packets as soon as they have k packets among the n ones [k information packets+$(n-k)$ redundant ones]. Their main drawback is that, in practice, they have a quadratic encoding and decoding complexity.

The other important classes of erasure codes are the rateless (Shokrollahi, 2003) and low-density-parity-check (LDPC/LDGM) codes (Roca *et al.*, 2003). They have better encoding and decoding complexity but they have a slightly lower erasure correction capability than MDS codes (especially for short length codes). For this reason, we only consider MDS codes in this paper.

Packet transmissions in networks often require a systematic erasure code, i.e. the packets to be protected are not modified by the encoder. It follows that the generator matrix must contain the $k\times k$-identity matrix. The construction of the non-systematic matrix can be done either by Vandermonde (Rizzo, 1997; Lacan *et al.*, 2005; Lacan and Fimes, 2004) or by Cauchy matrices (Bloemer *et al.*, 1995).

Since the Vandermonde-based generator matrix required a multiplication of two matrices (Lacan and Fimes, 2004), we present our construction with a Cauchy matrix.

A square Cauchy matrix is built from two vectors $(a_i)_{i=0}^{r-1}$ and $(b_j)_{j=0}^{r-1}$ such that a_i, b_i are $2r$ distinct elements defined over the finite field $\boldsymbol{F}$. A Cauchy matrix $\boldsymbol{A}$ is defined as follows:

$$\boldsymbol{A}=\left(\frac{1}{a_i-b_j}\right)_{i,j=0}^{r-1}.$$

The determinant is equal to:

$$\det(\boldsymbol{A})=\frac{\prod\limits_{0\le i<j\le r-1}(a_i-a_j)(b_i-b_j)}{\prod\limits_{0\le i<j\le r-1}(a_i+b_j)}.$$

This determinant is not equal to zero. It follows that the matrix is invertible. One important property is that with the same argument, it can be shown that any square submatrix of a Cauchy matrix is also invertible.

Since there exists efficient algorithms for Cauchy matrices to perform the inversion (in $O(r^2)$) and the multiplication with a vector (in $O(r\log^2 r)$), the Cauchy matrices are very good solutions for building systematic MDS codes.

CONSTRUCTION OF AN ERASURE CODE INTEGRATING DATA DEPENDENCIES

General principle

In systematic MDS (n, k) codes each redundancy packet is built from a combination of all the k source packets. Hence, receiving any packets subset containing at least k out of n encoded packets is sufficient to rebuild all the source packets ($n>k$). However, if the number of received packets does not reach the k threshold, none of the lost packets will be recovered, no matter their importance for the above application.

From this statement of fact, our goal was to give to some packets subsets built from the k source packets the possibility to be recovered even if the number of lost packets exceeds $r=n-k$. The idea behind this goal was to generate, for a systematic code, redundancy packets built only from a subset m of the k source packets, with $m<k$. Therefore, retrieving a source packet from these redundancy packets requires only $m-1$ additional encoded packets instead of the $k-1$ ones usually required in a classical MDS (n, k) code. The generator matrix which implements such a code is based on the generator matrix of a classical systematic MDS code on which we substitute some coefficients for the null element 0 in order to withdraw the contribution of the corresponding source packets in the construction of the redundancy packets.

To estimate roughly the consequence of the insertion of zero elements in the generator matrix, we performed short simulations where the zero elements were inserted in several determined positions. The initial code was a systematic MDS (24, 16) code where the non-systematic part of the generator matrix was built as a Cauchy Matrix (See Section 2). In Configuration 1, the zeroes were randomly distributed over the whole Cauchy matrix. In Configuration 2, the zeroes were placed in the first column and in the last Configuration 3, the zeroes were placed in the last row. Since the code is built with 33% of redundant packets, we simulated a channel with an average loss rate of 35%. For 50000 emitted code words of packets for each configuration, the received words of packets were decoded and the number of decoded information packets is considered. The results are presented in Table 1.

Table 1 Number of decoded packets

Configurations	Number of decoded packets
MDS	635193
1	−199
2	53
3	8121

Furthermore, from the last example and simulations, we deduced that the construction of the generator matrix which implements the idea expressed above will be closely related to the source data properties and to their exact location in the stream to be sent on the communication channel. Therefore, we directly worked on multimedia data.

Frame dependencies in video compression standards

Before sending them on a communication channel, multimedia data are first converted from analogue format to digital format through sampling, then compressed following a source coding scheme. This compression step, in addition to reduce transmitted data volume, builds up dependency relationships within data from which a hierarchical structure appears.

In the case of MPEG video coding, the coded stream contains three frame types, namely I, P and B. Each I frame is intracoded and can be consequently displayed independently. A P frame is coded relatively to the previous reference frame (I or P) and hence needs this reference frame to be displayed correctly. Finally, a B frame is a bidirectionally coded frame and needs the pair of previous and following reference frames (I or P) to be displayed. From these relationships that hold within frame types, it is clear that the effect of a packet loss that would affect an I frame or a P frame will propagate to all the following frames until the next correctly received I frame. However, if the packet loss affects a B frame, it will not affect another frame but the concerned B frame. Consequently, a GOP (Group of Picture) consists of different importance data, in which the I frames are the most important, followed by the P and the B frames.

Generator matrix construction

As noticed above, this importance order is nothing but the result of the dependency relationships that link the different frames. Hence, an efficient way to take into consideration the different importance levels of these data would be to integrate these dependency relationships in the construction of the redundancy packets. This integration is done by applying a simple set of rules to each generated redundancy packet. These rules are as follows:

A redundancy packet protecting a packet belonging to the frame *t* must: (a) protect all the packets belonging to this frame; (b) protect all the frames (i.e., all the packets belonging to those frames) on which this frame depends.

Considering the dependency relationships that hold within the different MPEG frames, applying the above defined rules to these data generate several kind of redundancy packets.

(1) The *rddce_I* type: Packet protecting the I frame.

(2) The *rddce_IP* type: Packets protecting a P frame and all the precedent P frames until the first I frame.

(3) The *rddce_IPB* type: Packets protecting a B frame, the corresponding pair of reference frames and all frames on which they depend.

Therefore, we can distinguish among the redundancy packets three regions delimited with two variables, *r_I* and *r_P*, where the first *r_I* packets are *rddce_I* type, the next *r_P* packets are *rddce_IP* type and the last packets are *rddce_IPB* type. Fig.1 summarizes this.

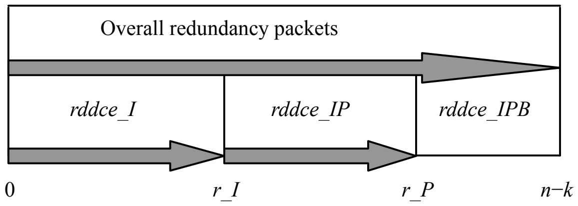

Fig.1 Redundancy packets divided into three types

However, since the number of redundancy packets is limited, the last *rddce_IPB* packets are rarely implemented as defined above and are instead set to protect all the frames.

Let us build redundancy packets to unequally protect a GOP which has the following pattern:

I B B P B B P B B

Each frame of this GOP is decomposed in packets of the same size. Assume the number of packets required for each of them is as described in Table 2.

Table 2 Number of packets required for each frame

Frame	Number of packets
I	5
B	1
B	1
P	2
B	1
B	1
P	2
B	1
B	1

The total number k of source packets is equal to 15. Let us now protect these packets with $n-k=5$ redundancy packets from which we assign $r\_I=2$ packets to build the *rddce_I* packets, another $r\_P=2$ packets to build *rddce_IP* packets and reserve the last packet to build *rddce_IPB* packet. The matrix generator for such a code is illustrated in the matrix $G_{20,15}$, where the $\alpha_{i,j}$ coefficients ($i\in\{1,\ldots,15\}$, $j\in\{1,\ldots,5\}$) are non-zero coefficients derived from a size-equivalent Cauchy matrix, k=15, n=20, $rddce\_I$=2, $rddce\_IP$=2, $rddce\_IPB$=1.

$$G_{20,15}=\left[\begin{array}{cccccc} & \alpha_{1,1} & \cdots & \cdots & \cdots & \alpha_{1,5} \\ & \vdots & & & & \vdots \\ & \alpha_{1,1} & \cdots & \cdots & \cdots & \alpha_{5,5} \\ & 0 & 0 & 0 & 0 & \alpha_{6,5} \\ & 0 & 0 & 0 & 0 & \alpha_{7,5} \\ & 0 & 0 & \alpha_{8,3} & \alpha_{8,4} & \alpha_{8,5} \\ I_{15} & 0 & 0 & \alpha_{9,3} & \alpha_{9,4} & \alpha_{9,5} \\ & 0 & 0 & 0 & 0 & \alpha_{10,5} \\ & 0 & 0 & 0 & 0 & \alpha_{11,5} \\ & 0 & 0 & 0 & \alpha_{12,4} & \alpha_{12,5} \\ & 0 & 0 & 0 & \alpha_{13,4} & \alpha_{13,5} \\ & 0 & 0 & 0 & 0 & \alpha_{14,5} \\ & 0 & 0 & 0 & 0 & \alpha_{15,5} \end{array}\right]$$

Application to video transmission over the packet erasure channel

A direct application of the proposed code is the protection of video transmission over a packet erasure channel. A simple way of implementation is to encode data on a GOP base, i.e. to each GOP is assigned a certain amount of redundant data. Beside, as we can see in the above example, the shape of the generator matrix is closely related to the size of each frame of the GOP. Hence, knowing that the size of a frame varies from GOP to GOP, the use of this code with real MPEG-video data requires dynamically building of the generator matrix for each GOP of the video sequence.

1. At the encoder side

In order to work the algorithm needs to know the following details a priori:

(1) The GOP frame pattern;

(2) The number of packets into which each frame has been decomposed;

(3) The total number of redundancy packets ($r=n-k$);

(4) r_I, the number of $rddce\_I$ packets;

(5) r_P, the number of $rddce\_IP$ packets.

One should notice that the last three variables are fixed by the user following the channel state (loss rate) and the error protection level assigned to the different frame type (r_I and r_P), while the two first depend exclusively on the characteristics of the source codec in use and the size of the network packet.

Once these variables are set or calculated, the encoder calculates the location of the coefficients to be set to zero in the generator matrix, and then build a classical systematic MDS generator matrix in which the coefficients pointed out above are put to zero. From this step on the encoder carries on classically.

2. At the decoder side

Since the generator matrix is dynamically built, the decoder has to know the exact generator matrix used to build the set of packets to be decoded. This could be done in several ways such as sending the location of the zero coefficients of the matrix along with the data. However this issue is not treated in this paper.

Assuming the decoder is somehow aware of the generator matrix, the decoding process works as described below:

Step 1: Rearrange the received packets in such a way as to replace the source lost packets by the redundant received packets. In the case where extra redundant packets are received, choose the one containing information about the source lost packet to be recovered.

Step 2: Rebuild the sub-matrix corresponding to the above rearranged packets.

Step 3: Inverse matrix:

(1) Since the generator matrix is not a Cauchy matrix any more, not every square sub-matrix of the generator matrix is invertible. However, there could exist square sub-matrixes of the rebuilt matrix that are invertible.

(2) From the matrix rebuilt in Step 2 find the larger square invertible sub-matrix. This is done by applying a Gauss elimination algorithm on the rebuilt matrix.

Step 4: Reconstruct the recovered packets. This step is done in parallel with the matrix inversion step.

SIMULATION RESULTS

In order to assess the performance of the proposed code and to compare it to other existing ones in terms of video end-user quality, we needed some complementary tools which would take care of all intermediates steps (except of course channel coding) from the original video in rawdata format to the reconstructed video also in rawdata format. To achieve this, we used the FFMPEG codec (http://ffmpeg.sourceforge.net/index.php) and the Evalvid Video Transmission and Quality Assessment Framework (Klaue *et al*., 2003). While FFMPEG was used to convert rawdata video into an MPEG4 stream and back again into rawdata, Evalvid framework was used to packetize the MPEG4 stream, send it on the network, generate a damaged received stream and finally calculate the PSNR and MOS quality metrics.

All the simulations were made on CIF-size "Foreman" video containing 300 frames decomposed into 13 GOPs with maximum number of 25 frames each. Within a GOP 2 B-frames were introduced between each pair of reference frames.

In the first part of the simulation, we set the total redundancy ratio per GOP to 25% and tried different configurations of the couple (*r_I*, *r_P*). For each couple we built a code and assessed its behaviour for loss rates varying from 4% to 40%, as shown in Fig.2.

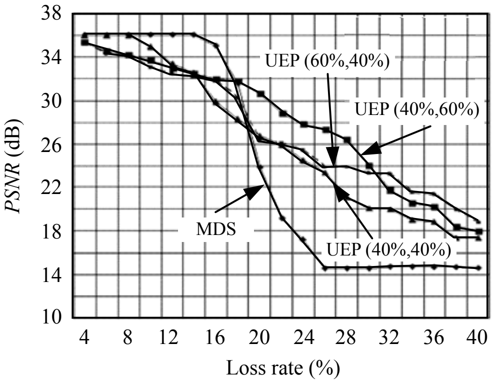

Fig.2 UEP vs MDS

The best performances are reached either by the MDS or UEPs codes depending on the value of the loss ratio *p* in comparison to the redundancy ratio *r*=20%. As a matter of fact, until *p*=16%, the best performances were reached by the MDS code, which behaves optimally by retrieving all the lost packets until *p*=14%, whereas the set of the UEPs codes are non-optimal and their *PSNR* curves are at *p*=16% [UEP (40%, 60%)], 3.08 dB lower than the MDS one. However, from this loss ratio on (*p*>16%), the MDS code is no more optimal and its performances drop abruptly, whereas the UEPs code, by protecting the most important data, withstands in a better way the loss ratio increases and achieves [for *p*=30% and the UEP (40%, 60%)] 14.12 dB PSNR gain over the MDS code. We can also notice that the UEP (40%, 60%) has globally the best performance among all the UEPs code. However, the proposed code has the peculiarity of being very flexible. Indeed, the couple (*r_I*, *r_P*) allows us to modify the code in order to fit the variable work conditions (loss rate). For instance, the UEP (0%, 0%) code is nothing but an MDS code.

In the second part of the simulation, we compared the performance of the UEP (40%, 60%) code with a size-equivalent reference Priority Encoding Transmission (PET) code (Albanese *et al*., 1996). The PET code we simulated was adapted from (Leicher, 1994). The MPEG stream is decomposed in three classes with different priorities, the I-frames class, the P-frames class and the B-frames class. Table 3 shows each frame type with the encoded packets fraction needed to recover it.

Table 3 PET priority function

Frame type	Fraction needed (%)
I	60
P	75
B	90

Note that the overall redundancy ratio satisfying the priority rule shown in Table 3 will depend on the size of each frame and thus will vary from GOP to GOP. This particularity was taken into consideration when comparing the different codes, by assigning to the MDS and the UEP codes—on a GOP base—the same number of redundancy packets that would fulfil the priority function of the PET codes [in the same condition (size of I, P and B frames)]. Afterwards, the average redundancy ratio found was equal to 24.35%.

The curves in Fig.3 confirmed the results found in the first part of the simulation and showed that the UEP code, from *p*=10% on, clearly outperformed the PET code, and achieved for *p*=26%, 7.94 dB *PSNR* gain. Furthermore, when mapping PSNR values to MOS score via the conversion table presented in (Klaue *et al*., 2003; Ohm, 1999), we can see that the proposed code can maintain a Fair MOS video quality in a packet loss range 10% larger and good MOS video quality in a packet loss range 12% larger.

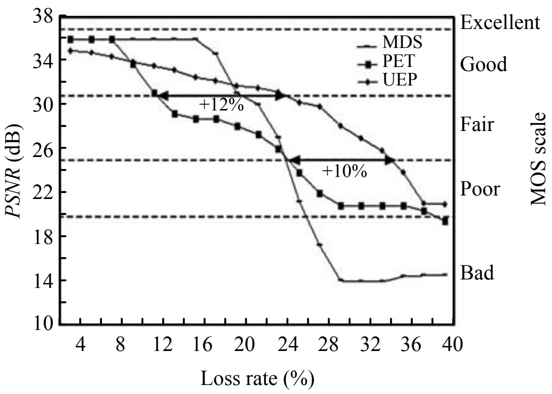

Fig.3 MDS, PET and UEP (40%,60%) codes with 24.37% average redundancy

CONCLUSION AND FUTURE WORK

In this paper, we presented a different approach for protecting multi-classes data by integrating with fine granularity existing dependencies. This integration was done by generating specific redundancy packets. Although the approach is based on very simple and intuitive rules, it showed very promising results validated by simulations.

Several improvements are planned for the proposed UEP erasure code. First, a more formal and theoretical interpretation of the parameters r_I and r_P must be done. The relation between these parameters, the distortion and the loss rate must be explored. Moreover, since in real channel, the loss probability varies, a good improvement would be to integrate the erasure code in a protocol providing feedbacks on the channel in order to dynamically tune the parameters r_I and r_P.

References

Albanese, A., Blömer, J., Edmonds, J., Luby, M., Sudan, M., 1996. Priority encoding transmission. *IEEE Trans. on Information Theory*, **42**(6):1737-1744. [doi:10.1109/18.556670]

Bloemer, J., Kalfane, M., Karpinski, M., Karp, R., Luby, M., Zuckerman, D., 1995. An XOR-Based Erasure-Resilient Coding Scheme. Technical Report ICSI TR-95-048.

Chou, P.A., Wang, H.J., Padmanabhan, V.N., 2003. Layered Multiple Description Coding. Proc. Int'l Packet Video Workshop. Nantes, France.

Klaue, J., Rathke, B., Wolisz, A., 2003. EvalVid—A Framework for Video Transmission and Quality Evaluation. Proc. of the 13th Internationnnal Conference on Modelling Techniques and Tools for Computer Performance Evaluation. Urbana Illinois, USA.

Lacan, J., Fimes, J., 2004. Systematic MDS erasure codes based on vandermonde matrices. *IEEE Communications Letters*, **8**(9):570-572. [doi:10.1109/LCOMM.2004.833807]

Lacan, J., Roca, V., Peltotalo, J., Peltotalo, S., 2005. Reed Solomon Error Correction Scheme. Work in Progress: <draft-lacan-rmt-fec-bb-rs-00>.

Leicher, C., 1994. Hierarchical Encoding of MPEG Sequences Using Priority Encoding Transmission (PET). Technical Report ISCI Lehrstuhl für Kommunikationsnetze Technische Universität München, TR-94-058.

Liebl, G., Wagner, M., Pandel, J., Weng, W., 2004. An RTP Payload Format for Erasure-Resilient Transmission of Progressive Multimedia Streams. Work in Progress: <draft-ietf-avt-uxp-07>.

Mohr, A.E., Ladner, R.E., Riskin, E.A., 2000. Approximately Optimal Assignment for Unequal Loss Protection. Conf. Image Processing. Vancouver, BC.

Ohm, J.R., 1999. Picture Signal Processing for Multimedia-System. Script, Institute for Communications and Theoretical Electrical Engineering (in German).

Rizzo, L., 1997. Effective erasure codes for relaible computer communication protocols. *ACM Review*, **27**:24-36.

Roca, V., Khallouf, Z., Laboure, J., 2003. Design and Evaluation of a Low Density Generator Matrix (LDGM) Large Block FEC Codec. Fifth International Workshop on Networked Group Communication (NGC'03). Munich, Germany.

Shokrollahi, A., 2003. Raptor Codes. Digital Fountain Inc., Tech. rep. DF2003-06-001.