Complexity Comparison of the Use of Vandermonde versus Hankel Matrices to Build Systematic MDS Reed-Solomon Codes

Ferdaouss Mattoussi, Vincent Roca, Bessem Sayadi

COMPLEXITY COMPARISON OF THE USE OF VANDERMONDE VERSUS HANKEL MATRICES TO BUILD SYSTEMATIC MDS REED-SOLOMON CODES

Ferdaouss Mattoussi[*] *Vincent Roca*[*] *Bessem Sayadi*[‡]

[*]Inria, France [‡]Alcatel-Lucent Bell Labs, France

{ferdaouss.mattoussi, vincent.roca}@inria.fr, {bessem.sayadi}@alcatel-lucent.com

ABSTRACT

Reed Solomon $RS(n, k)$ codes are Maximum Distance Separable (MDS) ideal codes that can be put into a systematic form, which makes them well suited to many situations. In this work we consider use-cases that rely on a software RS codec and for which the code is not fixed. This means that the application potentially uses a different $RS(n, k)$ code each time, and this code needs to be built dynamically. A light-weight code creation scheme is therefore highly desirable, otherwise this stage would negatively impact the encoding and decoding times.

Constructing such an RS code is equivalent to constructing its systematic generator matrix. Using the classic Vandermonde matrix approach to that purpose is feasible but adds significant complexity. In this paper we propose an alternative solution, based on Hankel matrices as the base matrix. We prove theoretically and experimentally that the code construction time and the number of operations performed to build the target RS code are largely in favor of the Hankel approach, which can be between 3.5 to 157 times faster than the Vandermonde approach, depending on the (n, k) parameters.

Keywords: AL-FEC; Reed-Solomon; Vandermonde matrix; Hankel matrix.

1. INTRODUCTION

Computer communications heavily rely on FEC codes to cope with data losses (e.g. IP datagram losses caused by congested routers in the Internet) and/or errors (e.g. during frame transmissions on a physical link). In particular systematic and Maximum Distance Separable (MDS) codes are rather attractive [10] for two reasons: first of all, because they achieve the maximum possible minimum distance for given length and dimension, which means that they feature optimal correction capabilities; but also because they are systematic, i.e. the information data is a part of the encoded data. In this work we focus on the erasure channel (rather than on the error channel). These FEC codes, that deal with data losses, are traditionally called Application-Level FEC (AL-FEC) codes since they are found in the upper transmission layers, often within the transport protocol or the application. However our code construction results could be applied in a similar way to codes for an error channel.

Reed Solomon (RS) codes are MDS erasure codes that can be put into a systematic form. Therefore their systematic generator matrix, G, can be written as: $G = [I_k|A_{k,n-k}]$, where I_k is the identity matrix of order k, and $A_{k,n-k}$ represents the repair matrix of order $k \times (n - k)$. The code will be MDS if and only if every square sub-matrix of A is non-singular [10][3].

In order to build this A matrix, it is of common practice to use Vandermonde matrices. However this approach is relatively costly as we will demonstrate. This is not an issue if the code is fixed, i.e. if its code dimension (k) and length (n) parameters are fixed and known in advance, since A can be pre-calculated in that case. For instance, when dealing with codes for the physical layer, the code is fixed and the codec implemented in hardware. But this assumption of a fixed code is no longer valid, in general, when dealing with the AL-FEC codes where the codec is usually a software component. In that case the $\{n, k\}$ parameters are dynamically determined, when there is a need to instantiate an AL-FEC encoder or decoder. This is also the case with GLDPC-Staircase codes [6], a class of Generalized LDPC codes that use a special form of systematic Reed-Solomon codes as component codes (i.e. the codes based on Hankel matrices introduced in this work). For such applications that generate Reed-Solomon codes on demand, dynamically, there is a clear interest in reducing the generator (G) matrix creation complexity.

[5] introduces a construction method for systematic MDS erasure codes, based on two Vandermonde matrices, which is the usual approach. In this work we focus on an alternative way of constructing the generator matrix, using Hankel matrices [8]. To the best of our knowledge, this is the first reference to this alternative way of designing systematic Reed-Solomon codes since their introduction in 1985 in [8]. We detail in this work how this is possible, we explain what complexity gains are expected during the systematic generator matrix creation stage, and finally we give an account of experiments carried out, using a C language software codec, in order to assess the practical gains.

This work was supported by the ANR-09-VERS-019-02 grant (ARSSO project) and by the Inria - Alcatel Lucent Bell Labs joint laboratory.

The paper is organized as follows. Section 2 details the construction methods for the Vandermonde and Hankel matrix variants. Then we compare their complexity in section 3. Finally, we conclude.

2. CONSTRUCTION METHODS OF SYSTEMATIC REED-SOLOMON CODES OVER $GF(Q)$

Let us first introduce the construction method for systematic MDS erasure codes.

2.1. Generalities

Systematic RS codes feature a generator matrix, G, of the form:

$$G = [I_k | A_{k,n-k}]$$

where A is a $k \times (n-k)$ matrix that does not contain any singular sub-matrix. Said differently, any square sub-matrix of A, formed from any i rows and any i columns of A, with $i \in \{1, \cdots, min\{k, n-k\}\}$, is non singular. Therefore the construction of a systematic RS generator matrix with the MDS property is equivalent to finding an appropriate A matrix. In this section we first detail how Vandermonde matrices are used to that purpose, then we detail an alternative solution based on Hankel matrices.

2.2. Codes based on Vandermonde matrices

A Vandermonde matrix $V(q,q)$ is defined by one vector of q distinct elements $(a_1, \cdots, a_q)$ over $GF(q)$, i.e. such that each a_i, $1 \leq i \leq q$, is an element of $GF(q)$. A representation of the Vandermonde matrix, $V = (a_i^{j-1})_{i,j}$ is:

$$V(q,q) = \begin{pmatrix} 1 & a_1 & \cdots & a_1^{q-1} \\ 1 & a_2 & \cdots & a_2^{q-1} \\ \vdots & \vdots & & \vdots \\ 1 & a_q & \cdots & a_q^{q-1} \end{pmatrix}$$

Vandermonde matrices defined over $GF(q)$ can contain singular square sub-matrices [3][5], and an upper bound of the number of singular sub-matrices is given in [1] [9]. Consequently these matrices cannot be directly used to design MDS systematic codes over $GF(q)$. However [3][5] introduce methods to use these matrices in order to build a systematic MDS RS generator matrix. Therefore, to obtain a systematic generator matrix $G(k,n)$ (and therefore a code $C(k,n)$), the simplest solution consists in considering two matrices: the first one is the $V(k,k)$ matrix formed by the first k columns of the second matrix $V(k,n)$, defined as previously. Then we invert the first matrix and multiply this inverse by $V(k,n)$. Clearly, the product $V(k,k)^{-1} * V(k,n)$ contains the identity matrix I_k on its first k columns, meaning that the first k encoding elements are equal to source elements. Besides, the resulting code features the MDS property. Therefore, the resulting systematic RS generator matrix based on Vandermonde matrix is equal to,

$$\begin{aligned} G &= V(k,k)^{-1} * V(k,n) && (1) \\ &= [I_k | V(k,k)^{-1} * V(k,n-k)] && (2) \\ &= [I_k | A(k,n-k)] && (3) \end{aligned}$$

This construction can be optimized by taking the representative vector of Vandermonde matrix $(a_1, \cdots, a_k)$ equal to $(1, a, a^2, ..., a^{k-1})$ where a is an element of $GF(q)$ with order k. Since the Vandermonde matrix is now defined by one element only (instead of q distinct elements), this matrix is denoted by $V(a)$ and has the following form:

$$V(a) = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & (a^1)^1 & (a^1)^2 & \cdots & (a^1)^{n-1} \\ 1 & (a^2)^1 & (a^2)^2 & \cdots & (a^2)^{n-1} \\ 1 & (a^3)^1 & (a^3)^2 & \cdots & (a^3)^{n-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & (a^{k-1})^1 & (a^{k-1})^2 & \cdots & (a^{k-1})^{n-1} \end{pmatrix}$$

This choice has two objectives: first, the inversion of $V(a)$ is easier since $(V(a))^{-1} = \frac{1}{k} \times V(a^{-1})$ [3]; secondly the matrix-vector multiplication between $V(a)$ and $V(a^{-1})$ can be performed very efficiently [2].

2.3. Codes based on Hankel matrices

Let us now focus on an alternative way of building systematic MDS generator matrix using Hankel matrix, a solution that has never been studied (as far as we can tell) since their introduction in [8]. Hankel matrices are square matrices whose values are constant along the ascending diagonals. The construction method is based on the creation of the maximal triangular array, B_q, defined over $GF(q)$, whose coefficients are constant along diagonals in a Hankel matrix fashion:

$$B_q = \begin{matrix} b_1 & b_2 & b_3 & \cdots & b_{q-2} & b_{q-1} \\ b_2 & b_3 & . & \cdots & b_{q-1} & \\ b_3 & . & . & \cdots & & \\ . & . & . & & & \\ . & . & b_{q-1} & & & \\ b_{q-2} & b_{q-1} & & & & \\ b_{q-1} & & & & & \end{matrix}$$

This triangular array has a "pure" Hankel matrix. The coefficients of the triangular array are equal to: $b_i = \frac{1}{1-y^i}$, where $1 \leq i \leq q-1$, y is an arbitrary primitive element of $GF(q)$ and y^i is computed over $GF(q)$. By using these coefficients, B_q has the property that every square sub-matrix is non-singular [8].

Next step consists in extracting from the upper triangle of B_q a rectangular sub-matrix A of size $k \times (n-k)$ (this is always feasible since $k \leq n < q$). This matrix has of

course the desired property that any square sub-matrix is non-singular. Therefore $A(k, n-k)$ can then be used to construct a generator matrix of a systematic RS code.

The following triangular array, T_q, is a particular case:

$$T_q = \begin{matrix} 1 & 1 & 1 & 1 & \cdots & 1 & 1 & 1 \\ 1 & b_1 & b_2 & b_3 & \cdots & b_{q-3} & b_{q-2} & \\ 1 & b_2 & b_3 & . & \cdots & b_{q-2} & & \\ 1 & . & . & \cdots & & & & \\ . & . & . & \cdots & & & & \\ . & . & b_{q-2} & & & & & \\ 1 & b_{q-2} & & & & & & \\ 1 & & & & & & & \end{matrix}$$

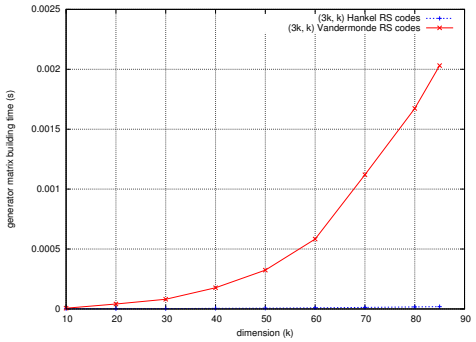
T_q has a "quasi" Hankel matrix form (instead of "pure" form as with B_q). It is built by removing b_{q-1} to B_q and adding an additional first row and column full of 1 entries. One can check that T_q does not contain any singular square sub-matrix as well [8]. Therefore one can extract an appropriate $A(k, n-k)$ matrix from the upper triangle of T_q as well in order to build the generator matrix of a systematic RS code. This second version has the nice property that the first repair symbol is also equal to the direct XOR sum of the source symbols (this is a key point of our GLDPC-Staircase codes, as detailed in [6]).

We therefore have two methods to build a generator matrix $G = [I_k|A_{k,n-k}]$ of a systematic MDS RS code: by extracting $A(k, n-k)$ from B_q ("pure" Hankel matrix form) or from T_q ("quasi" Hankel matrix form). With these techniques, *the systematic generator matrix is directly obtained by a trivial concatenating operation, instead of having to invert a matrix and then perform matrix-vector multiplications as is the case with the Vandermonde solution.* This is a major benefit when the code needs to be produced on-the-fly.

3. CODE CONSTRUCTION METHOD COMPLEXITY ANALYSIS

3.1. Performance evaluation environment

Let us now evaluate the generator matrix creation complexity, both in terms of the creation time and the number of elementary XOR operations. To that purpose, we are using a C language Reed-Solomon software codec for Vandermonde matrices, derived from L. Rizzo's well known codec[7]. This codec has been optimized and is freely distributed at:
`http://openfec.org/`
We also derived a codec for Reed-Solomon based on Hankel matrices that only differs during the generator matrix creation stage. The remaining of the two codecs, as well as the testing application, are strictly identical which warrants fair comparisons. In both cases we focus on $GF(2^8)$ codes since this is the most practical solution for software codecs: finite field elements are aligned on byte boundaries for easy transfers to/from memory buffers, and the finite field operations



(a) RS(3k,k) codes.

Fig. 1. Systematic generator matrix creation times.

Table 1. Generator matrix creation times and performance gains made possible by the Hankel approach.

	(30,10)	(250,50)	(250,100)	(250,125)
Vandermonde	0.007 ms	0.620 ms	2.577 ms	3.143 ms
Hankel	0.002 ms	0.011 ms	0.020 ms	0.020 ms
Ratio	3.5	56.36	128.85	157.15

pre-computed tables have a small size that fits well in CPU caches and/or RAM.

Note that we only consider the "pure" Hankel matrix variant, B_q. Using the "quasi" Hankel matrix variant, T_q, would slightly simplify the encoding (and perhaps decoding) steps, the first repair symbol being equal to a simple XOR sum of the source symbols.

Finally, all the initialization, encoding and decoding speeds are evaluated on a MacBookPro laptop, featuring a 2.4 GHz Intel Core i5 CPU, and running MacOS 10.6.7.

3.2. Generator matrix creation time

As explained in section 2.2, with the Vandermonde approach, the complexity of building G corresponds to the complexity of inverting $V_{k,k}$ and multiplying $V_{k,k}^{-1}$ by $V_{k,n-k}$. On the opposite, with the Hankel approach, once the B_q (or T_q with "quasi" Hankel variant) coefficients are computed, a simple concatenation is sufficient. In order to appreciate the practical consequences, we have measured these times.

Figure 1 illustrates the major gains permitted by the use of Hankel matrices, in terms of creation times. If the systematic generator matrix creation times increase exponentially with $RS(3k, k)$ codes based on Vandermonde matrices as k increases, the progression is linear with their Hankel equivalent. This is confirmed in Table 1 that evaluates the processing time gains made possible by the use of the Hankel approach: the gains vary between 3.0 and 51.87.

Table 2. Generator matrix creation complexity (number of XOR and table access (TA) operations) and performance gains made possible by the Hankel approach).

	(30,10)	(250,50)	(250,100)	(250,125)
Vandermonde	2,225 XOR	506,125 XOR	1,524,750 XOR	1,991,875 XOR
	2,706 TA	523,526 TA	1,569,551 TA	2,054,126 TA
Total # oper.	4,931 op.	1,029,651 op.	3,094,301 op.	4,046,001 op.
Hankel	30 XOR	250 XOR	250 XOR	250 XOR
	260 TA	10,500 TA	15,500 TA	16,125 TA
Total # oper.	290 op.	10,750 op.	15,750 op.	16,375 op.
Ratio	17.0	95.8	196.5	247.1

3.3. Generator matrix creation complexity

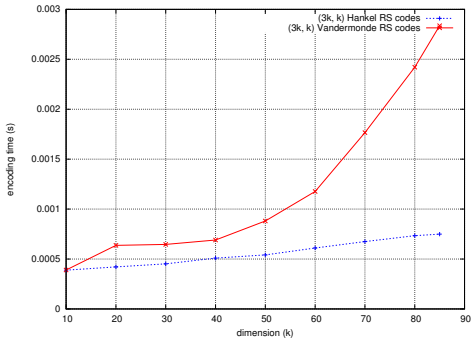
Let us now consider the number of operations required to build a systematic generator matrix of RS codes. With Vandermonde matrices, the creation of the systematic generator matrix requires finite field additions and multiplications over $GF(q)$ in order to perform matrix inversion and multiplication. Finite field additions consist in XORing the two values. Finite field multiplications are more complex, requiring in general a log table lookup, an addition operation and an exponentiation table lookup to determine the result. However, with $GF(2^8)$, multiplications can be pre-calculated and the result stored in a table of size 255×255. This is a common optimization with software codecs and this is how the initial Reed-Solomon codec was implemented. With this optimization, multiplying two elements of $GF(2^8)$ consists in accessing the right element of this pre-calculated table. Since this is a simple operation, the complexity evaluation only considers the number of XOR operations and ignores the number of multiplications.

As a result, for an $RS(n,k)$ code with a Vandermonde base matrix, the systematic generator matrix creation consists of :

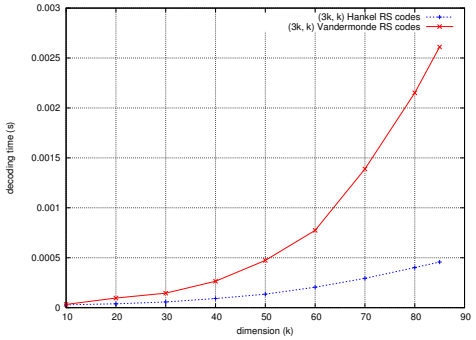
- the initialization of $V(a)$ which requires $k(n-1)$ read accesses to pre-calculated tables;
- a $(k \times k)$ matrix inversion, which requires $2k(k-1) + (k-1) + \frac{(k-1)(k-2)}{2}$ XOR operations, and $0.5(k-1)(k-2) + 2k(k-1) + 2k^2$ read accesses to pre-calculated tables;
- a $(k \times k)$ - $(k \times n-k)$ matrix multiplication, which requires $k(n-k)(k-1)$ XOR operations, and $k^2(n-k)$ read accesses to pre-calculated tables;

With Hankel matrices, the creation of the systematic generator matrix essentially consists in calculating the $b_i = \frac{1}{1-y^i}$ coefficients, with $1 \leq i \leq n-1$. Since the same values are used along the diagonals, we only calculate the n coefficients of the first line, which requires n XOR operations. In addition, the processing requires $2n + k(n-k)$ read accesses to pre-calculated tables.

We see that the Hankel approach outperforms the Vandermonde approach both in terms of the number of XOR operations and read accesses to tables. Table 2 shows the statistics obtained, counting the actual number of operations in the two codecs. The results confirm the benefit of the Hankel matrix approach, with speedups similar to that achieved in Table 1 when measuring time. The small difference is due to the simplification performed when counting operations: we do not include write operations, loop control operations, function call overheads, modulo calculations, and we assign the same cost factor to XOR and table access operations. A more detailed complexity analysis is feasible, but we consider that the accuracy achieved is sufficient to give an account of the behavior observed.



(a) Encoding time



(b) Decoding time

Fig. 2. Comparison between (3k,k) Vandermonde and Hankel Reed-Solomon codes during encoding and decoding.

3.4. Impacts on the global encoding and decoding times

Let us now try to answer to another question: what are the impacts of G creation on the total encoding or decoding times? Due to our assumptions, the systematic generator matrix creation time is included in the encoding (resp. decoding) times. Figures 2 show the relative gains made possible by the use of Hankel matrices on the encoding and decoding times, for several Reed-Solomon codes. These tests have been carried out with symbols of size 4 bytes each, i.e. the same operations are performed on each byte of the symbol (since erasures take place at the symbol level in case of Reed-Solomon codes for the erasure channel, see [4]). We see there is a clear gain in

using Hankel matrices, even if this gain depends on the actual code dimension and length being used.

However it should be noted that a symbol size of 4 bytes is rather small in case of AL-FEC codes. Symbols, that form the payload of UDP/IP datagrams, are more often on the order of a few hundreds of bytes. In that case, the relative benefit of reducing the matrix creation time would be reduced since the relative importance of data manipulations on symbols increases. The results of Figures 2 should therefore be regarded as upper bounds of the gains made possible by the use of Hankel matrices during encoding and decoding with many (but not necessarily all) AL-FEC codes.

4. CONCLUSIONS

Building a systematic MDS generator matrix for RS codes over GF(q) is equivalent to determining the A matrix of $G = [I_k|A_{k,n-k}]$, with the property that any square sub-matrix of A must be non-singular. In this paper, in addition to the traditional method for building A, based on Vandermonde matrices, we have introduced another approach, based on Hankel matrices. To the best of our knowledge, this is the first mention to this alternative way of designing systematic Reed-Solomon codes since their introduction in 1985, in [8]. We proved, both theoretically and experimentally, that this alternative solution is an order of magnitude simpler than the Vandermonde approach. The A sub-matrix of the generator matrix is produced immediately, instead of having to invert a matrix and multiplying this inverted matrix with another one. Major speedups, for instance up to 157, can be achieved thanks to the Hankel approach with our software C language Hankel Reed-Solomon codec, derived from a well-known Vandermonde Reed-Solomon codec for the erasure channel. This result is of high importance for all situations where a software $RS(n, k)$ codec needs to generate on the fly an RS code with appropriate dimension and length values.

5. REFERENCES

[1] J. Fimes, J. Lacan, "Estimation of the number of singular square submatrices of Vandermonde matrices defined over a finite field", Tech. Rep. ENSICA, no. RE-2003-01.

[2] I. Gohberg, V. Olshevsky, Fast algorithms with preprocessing for matrix-vector multiplication problems, Journal of Complexity, Vol. 10, December 1994.

[3] J. Lacan and J. Fimes, "A construction of matrices with no singular square submatrices", in Proc. International Conference on Finite Fields and Applications, pp.145-147, 2003.

[4] J. Lacan, V. Roca, J. Peltotalo, S. Peltotalo, "Reed-Solomon error correction scheme", IETF RMT Working Group, Request for Comments, RFC 5510, April 2009.

[5] J. Lacan, J. Fimes,"Systematic MDS erasure codes based on vandermonde matrices", IEEE Communications Lettres, Vol. 8, No. 9, September 2004.

[6] F. Mattoussi, V. Roca, B.Sayadi "Design of small rate, close to ideal, GLDPC-Staircase AL-FEC codes for the erasure channel", submitted for publication, 2012.

[7] L. Rizzo, "Effective erasure codes for reliable computer communication protocols", ACM SIGCOMM computer communication review, Vol. 27, No. 2, pp.24-36, April 1997.

[8] R. M. Roth, G. Seroussi, "On generator matrices of MDS codes", IEEE Transactions on Information Theory, Vol. IT-31, NO, 6, November 1985.

[9] I. Shparlinski, "On singularity of generalized Vandermonde matrices over finite fields", preprint, 2000.

[10] F. J. McWilliams, N.J.A. Sloane, "The theory of error correcting codes", North-Holland Publishing Company, ISBN: 0 444 85009 0, 1977.