Error-Correction in Two-Dimensional Fields

Jørn Justesen
COM
Technical University of Denmark
DK-2800 Kgs Lyngby, Denmark
Email: jju@com.dtu.dk

***Abstract*— We describe some of the properties of error-correcting codes that are specific to 2D applications. A construction is presented which combines Reed-Solomon component codes and a bipartite connection graph with excellent expansion properties.**

I. Introduction

Some information sources, storage media, and modes of transmitting information are essentially two-dimensional. Examples range from various advanced optical and nano-tech storage media to printed paper and computer screens. Some applications like documents, maps, or technical drawings may be thought of as a type of data bases. We use the terms 2D media to suggest that the distinction between sources and channels is not always natural.

The symbols in a 2D field do not have any natural total ordering. In some contexts they are read in a specific ('causal') order, i.e. line by line, but this is an arbitrary choice, and the content of the field may guide the user to access it in a different order.

In a single dimension, data are often organized in frames for the purpose of transmission or storage, in 2D we refer to a similar concept as a page. A page contains the equivalent of a frame header, i.e. a page number, a caption, a header/footer indicating the context, and a pattern serving to indicate the limits of the data an ensuring proper alignment. The limits of the code are typically chosen to coincide with those of the page.

The amount of data on a page is often large. However, the finite extent of the page is more noticeable since in the order $\sqrt{N}$ of N symbols are near one of the edges. This is in itself one of the reasons why stationary models are not as appropriate. In addition, documents are often composed of areas with significantly different properties, indicated by the lay-out of pages.

II. Properties of Error-Correcting Codes for 2D Fields

The impairments in the reading of a 2D medium include defects in the original surface, later degradation of the surface, imperfect alignment, and noise in the reading / transmission process. While the sources of the errors are not very different from the one-dimensional case, the large page size and lack of stationarity have led to choices of code structures that are different from standard transmission codes.

As examples of choices of codes we mention two cases: For video DVDs a product of two Reed-Solomon codes is used, $(208, 192, 17)$ and $(182, 172, 11)$ as part of an encoding process that also includes data compression and a run-length constrained binary code. For 2D labels of various kinds, the Data Matrix standard [1] specifies the encoding of square arrays of various sizes using Reed-Solomon codes. The codes are interleaved or shortened versions of codes over $F(256)$ correcting from 2 to 34 errors. Each symbol is mapped to a 3 by 3 configuration of black or white squares (with one corner missing) and packed diagonally into the area of the label. Both standards are supported by detailed specifications for alignment and inaccuracies of various kinds, which we shall not discuss in detail.

However, as different as these applications are, they show some common properties of error-correcting codes for 2D media:

- Reed-Solomon codes are used because of their robustness to clustered errors
- Long codes with relatively low decoding complexity are needed, leading to interleaved codes or product codes.
- A relatively large error-correcting capability is used due to the non-stationary nature of surface-related defects.
- In both cases the current standard replaces earlier codes with much more limited error-correcting capability.

Based on these observations we see the challenges of error-correction for 2D media to involve very long codes with limited decoding complexity and high reliability. In addition to the requirements that led to the codes in current use, the possibility of selectively decoding part of the data may become of interest. The following section describes a construction that meets these demands.

III. Composite Codes Defined by Bipartite Graphs

Error-correcting codes can be related to graphs in several ways. The approach taken here is related to a structure suggested by Tanner [2], and studied in detail by Barg and Zémor [3]. We assume that the graph is regular and bipartite with S right and left nodes and degree n. The graph should be a good expander. The symbols of the code are associated with the branches, and symbols that meet in a node have to satisfy the constraints of the component code. We shall consider only the case of (q, k, d) Reed-Solomon component codes over the field $F(q)$, although the field is most likely a binary extension field and symbols are represented as binary vectors.

A product code can be interpreted as a special case of such a code where the connection graph is a complete bipartite graph. Such a code has the advantage that the structure facilitates the decoding process by simplifying the accessing of relevant subsets of the positions and allowing a high degree of parallel processing.

Longer codes with decoding complexities essentially increasing linearly with the code length can be obtained by using connection graphs that have more nodes while the order of the nodes is kept constant. If the graph is generated in a random way or it is constructed to have good expansion properties, excellent performance is still predicted using the recent results on highly connected cores in random graphs [4].

The connection graph, A, will be derived from an S by S incidense matrix, M, for a suitable combinatorial structure. The bipartite graph is then described by the matrix

$$A = \begin{pmatrix} 0 & M \\ M' & 0 \end{pmatrix}$$

where 0 indicates an S by S zero matrix. For a regular graph, all rows and columns have n 1s, and the largest eigenvalue is clearly n. A good expander is characterized by having a second largest eigenvalue with absolute value of the order $\sqrt{n}$.

For a large code, the connection matrix, even in a sparse matrix format, provides an impractical way of performing the steps of the decoding algorithm, and it contains no suggestion about ways of indexing the symbols in order to allow the checking to be done in parallel. In the code described in the next section, we consider a particular arrangement of the code symbols, similar to the format of the product codes. Each component code operates on symbols within a single row or column, which significantly simplifies storage access. The connection graph has near optimal expansion properties.

IV. A Connection Graph from Generalized Quadrangles

In [6] we considered composite codes based on the incidence matrix of points and lines in a finite plane. With composite Reed-Solomon codes this approach leads to code lengths of the order q^3. To get codes approximating the parameters and performance that would be of real interest for 2D applications, we assume that the component codes are (q, k) and the composite codeword is a q^2 by q^2 array. It is preferable that the symbols of each code are confined to a single row or column. As indicated in the previous section, the performance of the code can be maintained with such relatively short component codes, and thus the structure has the potential of combining large pages with low decoding complexity. Using short component codes has the added potential of allowing decoding of a selected subset of information symbols, in particular if the number of errors in small.

We derive the required connection graph from a combinatorial structure known as a generalized quadrangle [5]. This structure consists of all points and a subset of the lines in projective 3-space over the field $F(q)$. Thus there are q^3+q^2+q+1 points, and the same number of lines is used. As the first step in the construction, a one-to-one linear mapping from a plane to a specific point in the plane is chosen. The lines are then chosen such that in each plane exactly the lines passing through the special point are included. In this way there are $q+1$ points on each line, and $q+1$ lines in the quadrangle pass through each point.

If the nodes of a bipartite graph are associated with the points (or planes) and the lines of the generalized quadrangle, the expansion property of the graph can be described in a simple way: From any point node, we can reach $q+1$ line nodes in a single transition. From these nodes, $q(q+1)$ new point nodes are reached in the following step. Finally in a third step we can reach exactly all remaining $q^2(q+1)$ line nodes. Similarly we can start from any line node and grow a tree that includes all point nodes at depth 3. The shortest cycles in the generalized quadrangles consist of 4 points and 4 lines, and thus the girth of the bipartite graph is 8.

The generalized quadrangle has more nodes than we need for the composite code, but we can get a graph with q^3 nodes on each side and exactly the required properties. The graph is reduced by removing the plane at infinity from the set of nodes. We do this in two steps by first selecting a point, P_0 in the plane, connecting it to the $q+1$ lines, and then taking all points on these lines. The remaining points and planes are then the corresponding Euclidean Geometry. Similarly we select a line, L_0, which passes through P_0, and remove this line as well as the $q(q+1)$ lines containing any point on it. The reduced bipartite graph is regular with degree q and has $S = q^3$ nodes.

The composite code defined on the reduced graph has length $N = q^4$, and the parity checks are defined by $2q^3$ component codes. Thus the rate is $R \geq 2k/q - 1$. An interesting lower bound on the minimum distance is given in [3], but if the code is decoded by iterated decoding of the component codes, the performance is not limited by the distance.

We want to arrange the codewords of the composite code as q^2 by q^2 arrays. In the projective version of the quadrangle, there are q^2+q+1 points in the plane at infinity, $q+1$ of which are on L_0. The remaining q^2 points are each connected to q lines in the reduced graph. Each such set of point nodes will represent the parity checks on a column in the array. Similarly we consider the q^2 lines in the projective quadrangle that do not contain P_0 and are not used in the reduced graph. These lines are connected to sets of q points, which define the rows of the array. In this way the symbol on a branch of the bipartite graph is assigned a row and column index.

Theorem: With the chosen labeling of the rows and columns, a unique symbol is assigned to each position in the array.

Proof: Assume on the contrary that two branches are assigned the same row and column index. We can then trace the paths back to the same point and line nodes defining these indices, and we obtain a cycle of length 6 contradicting that the girth of the graph is 8.

Thus the theorem shows that the code is represented by a square array where the symbols in each row and column are divided into q disjoint sets, each set being a codeword from the Reed-Solomon code.

V. The Second Eigenvalue of A

Since A is bipartite, the eigenvalues with largest absolute values are $\pm q$. The expander property of the graph is related to the next eigenvalue, which we find by considering MM'. Multiplying $(1, 0, 0, 0, ...)$ by this matrix we first obtain $(q, 1, 1, 1, ..., 0, 0, 0, ...)$ where the number of 1 entries is $q(q-1)$. In the bipartite graph this corresponds to starting at a particular node, making to transitions to q nodes on the other side, and then returning to the original node in q ways or going to $q(q-1)$ new nodes. Multiplying again by MM' we get two contributions to the first entry, q^2 from repeating the connections considered before, and $q(q-1)$ by returning from the new nodes reached in the previous stage. In order to get an eigenvector for the second eigenvalue, we start out from a balanced vector $(1, 0, 0, 0, ...) - (1, 1, 1, 1, 1, ..)/q^3$. Multiplying by MM' and normalizing, we can obtain the eigenvector by iteration. The entries in the resulting vector may be written as $((q-1)^2, (q-1), (q-1), ...)$ where $q(q-1)$ entries are $q-1$, and the remaining are $\pm(q-1)$ or ± 1. Multiplying this vector by MM' thus gives a first entry of $2q(q-1)^2$, and the eigenvalue is $2q$. For the original connection matrix A, the eigenvalue is the square root of this value, which is what we expect from a good expander graph.

VI. Conclusion

The construction presented in Section IV generalizes product Reed-Solomon codes in a way that would make them suitable for large 2D data sets. They offer a favorable combination of performance and low decoding complexity, including simple access to the relevant symbol sets for parallel processing. In addition it is possible to use iterative decoding of the component codes for recovering a subset of the data, in particular when the full error- correcting capability is not required.

A minimal example of such a code is a 256 by 256 array over $F(16)$, more realistic sizes would be $q = 64$ or $q = 256$.

References

[1] *International Standard ISO/IEC 16022*, ISO, 2000.

[2] M. Tanner,"A recursive approach to low complexity codes," *IEEE Transactions on Information Theory*, vol. IT-27, no. 5, pp. 533 - 547, Sep. 1981.

[3] A. Barg and G. Zémor,"Concatenated codes: serial and parallel," *IEEE Transactions on Information Theory*, vol IT-51, no. 5, pp.1625-1634, May 2005.

[4] B. Pittel, J. Spencer, and N. Wormald," Sudden emergence of a giant k-core in a random graph," *J. Comb. Theory*, Series B 67, pp. 111-151, 1996.

[5] C.T. Benson, "On the structure of generalized quadrangles", *Journal of Algebra*, vol. 15, pp.443-454, 1970.

[6] J. Justesen and T. Hoeholdt,"From concatenated codes to graph codes", *IEEE Information Theory Workshop 2004*, San Antonio, pp. 13-16, 24-29 Oct. 2004.