مجلة جامعة دمشق للعلوم الأساسية
مجلد 42-عدد 1 (2026): 49- 59

Damascus University Journal for the Basic Sciences
Vol 42 No.1 (2026): 49 -59

آلية لتعديل ترميزات الريد–مولر لتصحيح خطأ مضاعف

غصون أحمد عبد الكريم الجيرودي

أستاذ مساعد،عضو هيئة تدريسية في قسم الرياضيات، كلية العلوم، جامعة دمشق

ghussoun.aljeiroudi@damascusuniversity.edu.sy

الملخص:

تعد ترميزات الريد–مولر من الترميزات الشهيرة، وذلك لاستعمالاتها الواسعة في الهندسة الكهربائية والعلوم الحاسوبية. إن قابليتها للتطبيق تجعلها من الترميزات الأساسية في الاتصالات اللاسلكية. أستخدمت ترميزات الريد–مولر في أنظمة الاتصالات المتنقلة من الجيل الخامس بشكل كبير وفعال. بالإضافة إلى أنها لعبت دور جوهريا في الاتصالات عبر الفضاء. ولذلك لايزال فك ترميزات الريد–مولر يجذب العديد من الباحثين حتى الآن.

تسعى هذه المقالة إلى البحث في إمكانية تصحيح بعض الأخطاء الناتجة عن استخدامات هذه الترميزات، من خلال اقتراح تعديل لترميزات الريد–مولر الثنائية، والذي يسمح بتصحيح أخطاء الكلمة المستقبلة مرتين:

التصحيح الأول: الترميز المعدل سيؤدي إلى مصفوفة اختبار متكافئ سهلة الحساب، تستخدم لتصحيح بعض أخطاء الكلمة المستقبلة.

التصحيح الثاني: بعدما قمنا بالتصحيح الأول، تكون الكلمة أقرب أكثر للكلمة المرسلة، وبالتالي سيكون استخدام تقنية الأغلبية المنطقية أكثر فاعلية.

الكلمات المفتاحية: نظرية الترميز، ترميزات الريد–مولر، مصفوفة الاختبار المتكافئ، المصفوفات المتناثرة.

تاريخ الإيداع: 2023/10/12
تاريخ القبول: 2024/01/02

ISSN (online): 2789-6366
http://journal.damascusuniversity.edu.sy/index.php/index/index

Modified Reed-Muller Codes for Multiple Error Correction Approach

Ghussoun Ahmad Abdelkareem Al-Jeiroudi
Assistant Professor,Lecturer at Mathematics Department, Faculty of Sciences, Damascus University, Damascus, Syria, Specialty: Operational Research, ghussoun.aljeiroudi@damascusuniversity.edu.sy

Received: 12/10/2023
Accepted: 02/01/2024

Abstract

Reed-Muller codes (RM for shorten), are very popular in the domains of electrical engineering and computer science. Their versatility makes them essential, especially in the area of wireless communication. Within the context of 5G mobile networks, Reed-Muller codes contribute significantly to optimizing data transmission efficiency and reliability. Moreover, these codes play a crucial role in deep-space communication. Thus, the decoding of RM codes still attracts the researchers till now. This paper aims to decode these codes, by providing a modification to binary RM codes. The resulting code allows us to correct the receiving word twice:
The first correction approach: The modified code yields an easily compute parity check matrix, which is used to correct errors in the received word.
The second correction approach: after we correcting the errors with the previous approach, the word will become much closer to the sent word, and now the majority logic technique will be much more useful.

Keywords: Coding Theory, Reed-Muller Codes, Parity Check Matrix, Sparse Matrices.

1. Introduction

In 1954, mathematician David E. Muller introduced Reed-Muller codes making them one of the oldest code families. Irving S. Reed later developed the first efficient decoding algorithm, which are linear block codes. See (Meyer. L, 2021, page 12). However, RM codes quiet old codes but they exhibit excellent performance, particularly in applications like wireless and deep-space communication. Notably, first-order Reed-Muller codes have proven valuable in deep-space data transmission, such as the transmission of pictures from Mars (Meyer, 2021).

The popularity of these codes makes many researchers introduce decoding techniques to these codes. In this paper we provide a new approach to decode these codes.

In this paper we deal with binary RM codes. The generator matrix of these codes is sparse. We use this fact to construct a block upper triangular matrix as a generator matrix for our modified code. Since, we design a new algorithm (MG Algorithm) to construct the generator matrix for the modified code. Again we use the sparsity fact of the modified generator matrix, we use the sparse row-linked list, and we do Gaussian elimination. That is to construct an equivalent generator matrix in standard form. That leads to an easy parity check matrix, which is used to correct some errors of the receiving word. Afterwards, we do re-permutations to the correcting receiving word, and then we use the usual majority logic technique to correct more errors. These double error corrections, which we have provided, will be beneficial to many noise channels that require accuracy.

2. Literature Review

The RM codes are very popular, that makes many researches tackle the issue of decoding these codes. They are trying to provide faster techniques that are able them to correct as many errors as possible.

The decoding method used in (Cooke. 1999) is not proficient, but is quite simple to apply. As the encoding (generator) matrix and majority logic are used to determine whether that row was used in structure the receiving word. In (Key et al, 2016) the first-order and the second-order RM codes, R(1, m) and R(2, m), are used for permutation decoding. In the following papers (Santi et al. 2018) and (Kamenev. 2021) provide iterative decoding algorithms. The paper (Mengke et al. 2020) provides a decoding algorithm on factor graph with redundant code constraints that for high rate RM codes. The approach in (Fathollahi, 2021) consists of multiple sparse recursive projection aggregation, which is generated by performing only a selection of projections in each decoder. The authors in (Cho et al. 2022), applied an auto-decoder machine learning technique to get maximum likelihood decoding.

In section 3, we give an overview of binary Reed-Muller codes R(r, m). In section 4, we present a reminder on some sparse forms to save matrices. In section 5, we are going to construct the modified R(r, m) codes, and we provide an efficient algorithm to create the generator matrix of the modified code. Afterwards in section 6, we provide the two decoding approach.

3. Reed-Muller Codes:

The linear code C is a [n, k, d] code, where n is the length of codeword, d is the minimum distance of C and k is the dimension of C.

Definition: the definition of the minimum distance d for a code C is given by

$$d = \min\{d(x,y); x,y \in C, x \neq y\}$$

Where $d(x,y)$ equals the number of different coordinates of the codeword x and y (Hamming distance). (Henk C., 1993).

RM is a linear Code needs two integer parameters r and m where; $0 \le r \le m$. This code is denoted by R(r, m) and it is called Reed-Muller code of order r. The binary R(r, m) is linear code over the field F_2 defined by the function:

$$f(X_1, X_2, ..., X_m) \to \langle f(\alpha) \rangle |_{\alpha \in F_2^m}$$

That applies to the domain of all polynomials in $F_2[X_1, X_2, ..., X_m]$ of total of degree $\deg(f) \le r$.

Then, the binary R(r, m) code is defined:

$$R(r,m) = \{\langle f(\alpha) \rangle_{\alpha \in F_2^m} : f \text{ has total degree} \le r\} \qquad (1)$$

See (Guruswami 2010, chapter 8).

The binary R(r, m) code has length $n = 2^m$, its minimum distance is $d = 2^{m-r}$, and its dimension is $k = k(r,m)$ where:

$$k = k(r,m) = \sum_{i=0}^{r} \binom{m}{i} = 1 + \binom{m}{1} + ... + \binom{m}{r}, \qquad (2)$$

where $\binom{m}{k} = \frac{m!}{k!(m-k)!}$.

Thus the R(r,m) code is a code $[n,k,d] = [2^m, k(r,m), 2^{m-r}]$. (3)

See (Guruswami 2010, chapter 8).

The G generator matrix of R(r, m) has the dimensions $k \times n$. The rows of the G generator matrix for R(1, m) are given by:

$$G = \{v_0, v_1, ..., v_m\}.$$

The rows of the G generator matrix for R(r, m), where $r > 1$, are given by:

$G = \{v_0, v_1, ..., v_m, v_1 v_2, ..., v_1 v_m, .. v_{m-1} v_m, ...$up to products of degree $r\}$, Where $v_0 = (1,1,...,1)$, and for

$$v_i = (\underbrace{0,...,0}_{2^{i-1}}, \underbrace{1,...,1}_{2^{i-1}}, ..., \underbrace{0,...,0}_{2^{i-1}}, \underbrace{1,...,1}_{2^{i-1}})$$

(4)

The product, which is defined above, is given by:

$$x.y = (x_1 y_1, x_2 y_2, ..., x_n y_n).$$

See (Cooke, 1999).

Example: Let R(2, 4), which is a [16, 11, 4] code. Its generator matrix G is given by:

$$G = \{v_0, v_1, v_2, v_3, v_4, v_1 v_2, v_1 v_3, v_1 v_4, v_2 v_3, v_2 v_4, v_3 v_4\}. \qquad (5)$$

That gives:

$$G=\begin{bmatrix}1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1\\0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1\\0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1\\0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1\\0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1\\0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1\\0&0&0&0&0&1&0&1&0&0&0&0&0&1&0&1\\0&0&0&0&0&0&0&0&0&1&0&1&0&1&0&1\\0&0&0&0&0&0&1&1&0&0&0&0&0&0&1&1\\0&0&0&0&0&0&0&0&0&0&1&1&0&0&1&1\\0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1\end{bmatrix}$$

4. Sparse Forms:

The generator matrix G of binary R(r, m) is a spare matrix. We use this fact to construct a modification to G, which we will call it $\widetilde{G}$ that gives the modification code $\widetilde{C}$. Before we write the algorithm to construct $\widetilde{G}$, we provide a quick remainder of saving spars matrices. There is various ways to save the spares matrices. In this paper we will use two types of matrix's saving. Since, in the first approach we focus on the columns layout of the elements (we use matrix's saving in column-linked list), and in second approach we do rows operations (we use matrix's saving in row-linked list).

4.1 The sparse matrix's saving in column-linked list:

The matrix is saved in three vectors; N_C_A , R_A and V_A.

Let A be $m\times n$ matrix, and let the vector s contains the number of nonzero elements in each column of A. Let Anz the number of nonzero elements in A. Then:

$$\textbf{N\_C\_A[1]=1, and} \quad \textbf{N\_C\_A[i+1]= 1+}\sum_{j=1}^{i} s[j], \qquad \textbf{(6)}$$

where $1\le i\le n$ the size of this vector equals to $n+1$.

The vector R-A consists of the orders of the rows in which the nonzero elements in columns exist. the size of this vector equals to Anz.

The vector of V_A consists of the value of nonzero element in columns order, the size of this vector equals to Anz. See (Duff et al. 1986, Chapter 2).

4.2 The sparse matrix's saving in row-linked list:

The matrix is saved in three vectors; N_R_A , C_A and V_A.

Let A be matrix $m\times n$, and let the vector s contains the number of nonzero elements in each row of A. Let the number of nonzero elements of A equals to Anz.Then:

$$\textbf{N\_R\_A[1]=1, and} \quad \textbf{N\_C\_A[i+1]= 1+}\sum_{j=1}^{i} s[j], \qquad \textbf{(7)}$$

where $1\le i\le m$ the size of this vector equals to $m+1$.

The vector C-A consists of the orders of the column in which the nonzero elements in rows exist. The size of this vector equals to Anz.
The vector of V_A consists of the value of nonzero element in rows order, the size of this vector equals to Anz. See (Duff et al. 1986, Chapter 2).
Example: Let the matrix

$$A=\begin{bmatrix}0 & 8 & 0 & 0 & 9\\ 2 & 3 & 0 & 7 & 0\\ 0 & 0 & 5 & 0 & 4\end{bmatrix}$$

The number of nonzero elements in the matrix A is $Anz=7$.
The matrix can be saved in either, column-linked list:
$N\_C\_A=[1,2,4,5,6,8],\ R\_A=[2,1,2,3,2,1,3],\ V\_A=[2,8,3,5,7,9,4],$
or can be saved in row-linked list:
$N\_R\_A=[1,3,6,8],\quad C\_A=[2,5,1,2,4,3,5],\ V\_A=[8,9,2,3,7,5,4].$

5. Modifie the R(r,m) codes:

In this section we will provide a new algorithm (MG) to construct the modified generator matrix $\widetilde{G}$. This matrix has the form: $\widetilde{G}=[U \mid B]$, where U is sparse upper triangular $k\times k$ matrix. In this algorithm we use permutation. The code $\widetilde{C}$, which is generated by $\widetilde{G}$, has the same [n, k, d] as R(r, m) code. As, columns permutation preserve the linear independently. Hence, the dimension k will be the same as R(r, m). In addition, the weight of the code words will not change by doing indexes permutation. Also, the distance between of any two code words will be unchanged too. Consequently, the minimum distance d will be the same in the codes $\widetilde{C}$ and R(r, m).

5.1 Construct the matrix $\widetilde{G}$:

We are going to save the generator matrix G of binary R(r, m) in sparse column-linked list, the vectors; N_C_G similar as above, the vector R_G is also equals to the row of nonzero elements in the columns order, but with one difference; we save the row number starting from the end of the column till its beginning. Furthermore, there is no need to the vector V_A (as all nonzero elements equal to one). In addition, we choose the block upper triangular U with minimum nonzero number of G columns. As, we choose the number of nonzero in columns' of U is less than or equals to (2^r). That gives a sparser matrix U.

Modified Generator Algorithm:
We store the matrix G in two vectors as we mentioned above. We choose the sparse column-linked because we will use columns permutation. The new order of elements R_G plays a crucial role to choose the diagonal elements straightforward without needing to loop over the elements of the column. In addition, the N_C_G will help to choose the columns of U with (2^r) elements at most (to get spare U). The structure of RM's generator matrix makes this achievable.

The vector column_p will give the new columns order to construct $\widetilde{G}$.

The MG Algorithm:

Let h=l=1, column_p=[1 2 ... n]
For i=1 to n
j= column_p[l] , l=l+1
If (R_G[j] = h) and

(N_C_G [j+1] - N_C_G [j] $\leq 2^r$) then

Swipe between column_p[h] and column_p=[j]
h=h+1, l=h
stop when h=k+1

This algorithm gives the new order of columns of G in order to construct $\widetilde{G} = [U \quad | \quad B]$.

In addition, this algorithm provides a sparse block upper triangular U. Furthermore, it is an efficient algorithm, as it is required only on loop over the columns of G, also it stops when k entry is found.

Example: The R(2, 4), is a [16, 11, 4] code, as we see in the previous example its generator matrix G. In order to store this matrix in the formal way we require 16×11=176 places to store. However, in our proposal way require 17 places to store N_C_G and 72 places to store R_G. Hence we require only 89 places to store G. Consequently, in our propose approach we save 86 places of the storage.

The matrix G is store in these vectors:

N_C_G = [1, 2, 4, 6, 10, 12, 16, 20, 27, 29, 33, 37, 44, 48, 55, 62, 73].

The vector R_G contains 72 elements we will mention only few elements from it:

R_G = [1, 2, 1, 3, 1, 6, 3, 2, 1, 7, 4, 2, 1, ……., 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1].

Besides, we will save more storage places for bigger generator matrices.

Example:

R(1,4) is a code of [16, 5, 8], its generator matrix is

$$G = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

The matrix G is saved in column-linked list, the vectors as the following;

N_C_G =[1, 2, 4, 6, 9, 11, 14, 17, 21, 23, 26, 29, 33, 36, 40, 44, 49]

R_G=[1,2,1,3,1,3,2,1,4,1,4,2,1,4,3,1,4,3,2,1,5,1,5,2,1,5,3,1,5,3,2,1,5,4,1,5,4,2,1,5,4,3,1,5,4,3,2,1]

The result of the MG algorithm is

column_p=[1, 2, 3, 5, 9, 6, 7, 8, 4, 10, 11, 12, 13, 14, 15, 16]. That give fast and sparser block upper triangular matrix U and, thus $\widetilde{G} = [U \quad | \quad B]$:

$$\widetilde{G}=\begin{bmatrix}1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1\\0&1&0&0&0&1&0&1&1&1&0&1&0&1&0&1\\0&0&1&0&0&0&1&1&1&0&1&1&0&0&1&1\\0&0&0&1&0&1&1&1&0&0&0&0&1&1&1&1\\0&0&0&0&1&0&0&0&0&1&1&1&1&1&1&1\end{bmatrix}$$

6. The double decoding approach:

6.1 The first decoding approach:

We use the MG algorithm to construct $\widetilde{G}$, which gives the modified code $\widetilde{C}$. We are going to use the maximum likelihood algorithm to calculate errors of the receiving word r which encode by the code $\widetilde{C}$. In order to do so we require to calculate the parity Check matrix of the code $\widetilde{C}$. First we write the matrix $\hat{G}$ which is an equivalent matrix to $\widetilde{G}$ (using rows operations only. the rows operations will not change the code. Hence, rows operations change only the basics). That is to get:

$$\widetilde{G}=[U \quad | \quad B]\cong[I_k \quad | \quad P]=\hat{G}$$

We save the matrix $\widetilde{G}$ in sparse row-linked list, and we do Gaussian elimination.

The number of row operations we need to get $\hat{G}$ less than or equals to $(2+\sum_{i=3}^{k}(2^r-1))$. Since we construct U with at most (2^r) elements in each columns.

If the generator matrix is in the standard form $[I_k \quad | \quad P]$, then its parity Check matrix will be $[p^T \quad | \quad I_{n-k}]$. See (Henk, 1993, , page 14).

Our approach yields straightforward parity Check matrix. Hence we construct $\hat{G}$ in the form $[I_k \quad | \quad P]$, its parity Check matrix is $H=[p^T \quad | \quad I_{n-k}]$.

One the parity Check matrix is computed, the maximum likelihood algorithm is used to calculate errors of the receiving word r (which is send by a noise channel), see (Henk 1993, page 16).

6.2 The second decoding approach:

The first approach corrects errors in the receiving word r, we will denote this correcting receiving word by $\tilde{y}$. Now we return this word $\tilde{y}$ into Reed-Muller code. We do re-permutation to the indexes of $\tilde{y}$. That is by doing inverse of columns permutation, which is done in the MG algorithm. That yields the word y.

Now we can use the Majority Logic Decoding to correct the remains errors in y. See (Guruswami et al. 2022, page 270) for more details on Majority Logic Decoding.

Example:

In the previous example the MG algorithm changes the columns by using
column_p=[1, 2, 3, 5, 9, 6, 7, 8, 4, 10, 11, 12, 13, 14, 15, 16].
Then $y = [\tilde{y}_1, \tilde{y}_2, \tilde{y}_3, \tilde{y}_9, \tilde{y}_5, \tilde{y}_4, \tilde{y}_6, \tilde{y}_7, \tilde{y}_8, \tilde{y}_4, \tilde{y}_{10}, \tilde{y}_{11}, \tilde{y}_{12}, \tilde{y}_{13}, \tilde{y}_{14}, \tilde{y}_{15}, \tilde{y}_{16}]$.
The Majority Logic Decoding is used to correct errors in y.
By doing the double decoding approaches we will get as close as possible to the sending word (original massage).
In the following example, we will put it all together. First, we will choose a codeword x. Secondly, we will introduce errors to it, (as if x has been sent by a noisy channel). This results in the receiving word r. Thirdly, we will use our approach to correct these errors.

Example:

R(1,4) is a [16, 5, 8] code, its generator matrix G .

We use MG Algorithm to got the matrix $\widetilde{G}$ as in the previous example, then we do row operations as we mention above to get an equivalent $\hat{G}$

$$\widetilde{G} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \cong$$

$$\cong \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} = \hat{G}$$

We got $\hat{G}$ by doing 3 rows operations. Its parity Check matrix H is given by

$$H = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Let the codeword (sending word) is $x = [0000000011\ 111111\ ]$
And let the receiving word is $r = [0110000011\ 111110\ ]$ (we made 3 errors, assuming they have happened by a noisy channel while the word is translated). We use our algorithm to permuted the receiving word to get

$\tilde{r} = [0110100001\,1111110\,]$. we use the parity Check matrix H in maximum likelihood algorithm (Henk 1993, page 16), we get the error vector
$e = [0110000000\,000001]$ by correcting these errors:
$\tilde{y} = \tilde{r} + e = [0000100001\,111111\,]$.

Now, we re-permutation to the indexes of $\tilde{y}$, that yields

$y = [0000000011\,111111\,]$ is one of R(1,4) codeword, so no further error correction is needed and y is the sending word.

In the previous example we use R(r,m) code with small r and m. However, our approach is proposed for large binary Reed-Muller codes, as we use sparse matrices.

7. Results and Discussion:

We provide a modification to RM codes, which allow us to double correct the errors of the receiving word. The generator matrix of RM is sparse so we use this fact to construct a sparse modified generator matrix, which is done by Modified Generator Algorithm. This approach corrects errors by constructing the parity check matrix. After that, majority logic technique is used to correct the remains errors in the receiving word. As, we have shown in the previous example. We want to draw attention to the users of decoding RM codes. The double decoding approach presented in this paper correct more errors compared to single decoding approach. However, single decoding approach will be much faster. Hence, the user can choose between accuracy and speed depending of their applications.

8. References

1. Cho H. and Song Y., 2022, High Speed Decoding for High-Rate and Short-Length Reed-Muller Code Using Auto-Decoder, Applied Sciences 12(18), 9225.
2. Cooke B., 1999, Reed-Muller Error Correcting Codes, MIT Undergaduate Journal of Mathematics 1(06), 21-26.
3. Duff I., Erisman A. and Reid J., 1986, Direct Methods for Sparse Matrices, CLARENDON Press. Oxford.
4. Fathollahi D., Farsad N., Hashemi S. and Mondelli M., 2021, Sparse Multi-Decoder Recursive Projection Aggregation for Reed-Muller Codes, IEEE International Symposium on Information Theory, 1082-1087.
5. Guruswami V., 2010, Introduction to Coding theory, 15-859V, Spring.
6. Guruswami V., Rudra A. and Sudan M., 2022, Essential Coding Theory, National Science Foundation under NSF CAREER grantCCF-0844796.
7. Henk C., 1993, Coding Theory a first course, Eindhoven University of Technology Netherlands.
8. Kamenev M., 2021, On Decoding of Reed-Muller Codes Using a Local Graph Search, IEEE Transactions on Communications 70(2), 739-748.
9. Key J., McDonough T. and Mavron V., 20016, Reed-Muller codes and permutation decoding, Institute of Mathematics and Physics Aberystwyth University, Aberystwyth, Ceredigion SY23 3BZ, U.K.
10. Mengke L., Hager C. And Pifister H., 2020, Decoding Reed-Muller Codes Using Redundant Code Constraints, IEEE International Symposium on Information Theory, 42-47
11. Meyer L., 2021, Coding and Decoding of Reed-Muller Codes, Faculty of Health, Science and Technology Mathematics.
12. Santi E., Häger C. and Pfister H., 2018, Decoding Reed-Muller Codes Using Minimum-Weight Parity Checks. IEEE International Symposium on Information Theory – Proceedings :1296-1300.