Weight Distribution of the Binary Reed-Muller Code $\mathcal{R}(4,9)$

Miroslav Markov and Yuri Borissov

Institute of Mathematics and Informatics, Bulgarian Academy of Sciences
miro@math.bas.bg, youri@math.bas.bg

Abstract. We compute the weight distribution of the $\mathcal{R}(4,9)$ by combining the approach described in D. V. Sarwate's Ph.D. thesis from 1973 with knowledge on the affine equivalence classification of Boolean functions. To solve this problem posed, e.g., in the book of MacWilliams and Sloane, we apply a refined approach based on the classification of Boolean quartic forms in eight variables due to Ph. Langevin and G. Leander, and recent results on the classification of the quotient space $\mathcal{R}(4,7)/\mathcal{R}(2,7)$ due to V. Gillot and Ph. Langevin.

Keywords: binary Reed-Muller code, weight distribution, affine equivalence

1 Introduction

For basic coding theoretical notions, we refer to [13]. All considered codes in this paper are binary, i.e., over the alphabet $\mathbb{F}_2 = \{0,1\}$.

The binary Reed-Muller codes form one of the oldest studied families of codes invented in 1950s and have an easy-to-implement decoding algorithm based on majority-logic circuits. However, there are few general results about their weight structure, i.e., the weight distributions are known only for:

- the 1^{st} and 2^{nd} order codes of that kind [17] (1970);
- arbitrary order when the weight $< 2d$ [8] (1970), and later on (in 1976) had been extended for weights $< 2.5d$ where d is the minimum weight [9];

Results on the weight spectra of some third order codes are presented in an earlier work [20], and on the spectra of whole families of binary Reed-Muller codes in the very recent work [2]. Some partial results concerning the weight distribution of the third and fourth order Reed-Muller codes are obtained in [15], [9], [18] and [19]. For more information about the weight distributions of binary Reed-Muller codes of particular lengths and orders, the reader is referred to [16].

The weight spectrum of the fourth order Reed-Muller code $\mathcal{R}(4,9)$ of length 512, has been found in [2] and presented as a numerical example which demonstrates the technique developed there. To our knowledge, there have been very few attempts to determine the (exact) weight distribution of this code, which was listed among the smallest Reed-Muller codes whose weight distributions were unknown (in 1977) (see, [13, p. 447]). Specifically, in the concluding remarks of his Ph.D. thesis [15],

D. V. Sarwate has discussed the applicability of the methods described by him to Reed-Muller codes of lengths larger than 256. He has estimated that there are too many equivalence classes of cosets from the desired type and has come into conclusion that enumerating the $\mathcal{R}(4,9)$ seems out of reach through them. Another promising way to attack the considered problem consists in using the fact that we are dealing with a double-even binary self-dual code and a general form of the weight enumerators of such codes is known from the work of A. M. Gleason (see, e.g., [13, Ch.19]). But, although this second approach has proven itself in the case of shorter codes of that kind and requires modest computational efforts, for its successful application one needs more intrinsic knowledge about the $\mathcal{R}(4,9)$ than those presented in [8] (see, [3, Ch. 11] for details).

This paper is organized as follows. In the next section we give the necessary preliminaries. In Section 3 a refined approach to the problem under consideration enabling one to save computational efforts is exposed. Some conclusions are drawn in the last section.

2 Preliminaries

For basic knowledge on Boolean functions and their applications in Coding Theory and Cryptography, we direct the reader to [1] and [4]. Herein, for the sake of completeness, we recall the classical definition of the binary Reed-Muller code.

Definition 1. *The r^{th} order binary Reed-Muller (or RM) code $\mathcal{R}(r,m)$ of length $n = 2^m$, for $0 \leq r \leq m$, is the set of all binary vectors $\mathbf{f}$ of length n which are truth tables of Boolean functions $f(\mathbf{x}), \mathbf{x} = (x_1, \ldots, x_m)$, having algebraic normal forms of degree at most r.*

Henceforth the binary vector $\mathbf{f}$ of length 2^m will be identified with corresponding Boolean function f in m variables.

In order to present our results we need to recall the definition of the weight distribution of a code, i.e., an arbitrary set $\mathbf{C}$ of vectors with fixed length n (this definition holds in particular for cosets of binary linear codes).

Definition 2. *The weight distribution of a code $\mathbf{C}$ of length n is the vector $W(\mathbf{C}) = (W_0, \ldots, W_n)$, where W_i denotes the number of codewords with Hamming weight i.*

Accordingly, we recall the definition of the simplest weight enumerator of a code.

Definition 3. *Weight enumerator of a code $\mathbf{C}$ possessing weight distribution $W(\mathbf{C}) = (W_0, \ldots, W_n)$ is defined as the following polynomial in the indeterminate z:*

$$\mathcal{W}[z; \mathbf{C}] = \sum_{i=0}^{n} W_i z^i.$$

In this paper, we make use of two facts for the first time exposed in [15] and stated in the next two theorems. (For $0 \leq r \leq m$, the set of all homogeneous polynomials on m binary variables of algebraic degree r adjoined with the 0 is denoted by $\mathcal{H}^{(r)}(m)$.)

Theorem 1. *([15, 5.12]) For $0 \leq r \leq m$, the following holds:*

$$\mathcal{W}[z; \mathcal{R}(r+2, m+2)] = \sum_{p \in \mathcal{H}^{(r+2)}(m+1)} \mathcal{W}^2[z; p + \mathcal{R}(r+1, m+1)].$$

Theorem 2. *([15, 5.13]) Let $p = e + f x_{m+1}$, with given $e \in \mathcal{H}^{(r+2)}(m)$ and $f \in \mathcal{H}^{(r+1)}(m)$. Then the weight enumerator of the coset $\mathcal{C}(p) = p + \mathcal{R}(r+1, m+1)$ equals to:*

$$(*) \quad \sum_{g \in \mathcal{H}^{(r+1)}(m)} \mathcal{W}[z; e + g + \mathcal{R}(r, m)] \cdot \mathcal{W}[z; e + g + f + \mathcal{R}(r, m)].$$

For definition of the general affine group $GA(m)$ and its subgroup the general linear group $GL(m, 2)$, we refer to [13, Ch.13.9]. The action of $A \in GA(m)$ on a Boolean function $f(\mathbf{x})$ will be denoted by $f \circ A$, i.e., $f \circ A = f(A(\mathbf{x}))$. Another necessary definition is that of affine equivalence of two cosets of Reed-Muller code:

Definition 4. *The cosets C_1 and C_2 of $\mathcal{R}(r, m)$ with representatives $f_1 \in C_1, f_2 \in C_2$, respectively, are called affine equivalent if there exist a transformation $A \in GA(m)$ such that $f_1 \circ A = f_2$.*

In this article, we extensively use the following well-known property (see, e.g., [7]):
Property $\mathcal{P}$. *The weight enumerators of two affine equivalent cosets of a Reed-Muller code are identical.*

Affine equivalence classification of the cosets of RM codes is useful in studying important coding theoretical and cryptographic properties of Boolean functions comprising them. A strategy how to compute the complete classification of Boolean quartic forms in eight variables, i.e., the classification of the quotient space $\mathcal{R}(4, 8)/\mathcal{R}(3, 8)$ under the action of $GL(8, 2)$, is presented in [12]. Here, just as an extract of this result, we point out that the Boolean quartic forms of eight variables can be classified in 999 (see, as well [7]) linear equivalence classes listed in [10]. Recently, the interest in that topic has been renewed by [5] which (among other things) provides affine equivalence classification of the quotient space $\mathcal{R}(4, 7)/\mathcal{R}(2, 7)$. The authors of [5] and [12] have also outlined applications of their results concerning the covering radii of some RM codes, and Boolean functions in the family of bent ones. In Section 3, we point out yet another application, namely, computing the weight distribution of $\mathcal{R}(4, 9)$.

3 The refined approach

3.1 Rationale

Now, we describe a strategy following which makes feasible the computation of interest.

For $1 \leq r \leq m$, let $n(r, m)$ be the number of $GL(m, 2)-$orbits in the quotient space $\mathcal{R}^*(r, m) = \mathcal{R}(r, m)/\mathcal{R}(r-1, m)$. Also assume that an arbitrary numbering of these orbits (linear equivalence classes) has been fixed.

Corollary 1. *Let $p_i \in \mathcal{H}^{(r+2)}(m+1)$ be a representative of the i^{th} linear equivalence class in $\mathcal{R}^*(r+2, m+1)$ with size L_i. Then the following holds:*

$$\mathcal{W}[z; \mathcal{R}(r+2, m+2)] = \sum_{i=1}^{n(r+2,m+1)} L_i \mathcal{W}^2[z; p_i + \mathcal{R}(r+1, m+1)]. \qquad (1)$$

Proof. The assertion is an immediate consequence of Theorem 1 and Property $\mathcal{P}$. □

The above corollary reduces the number of needed weight enumerator computations to the number $n(r+2, m+1)$ which is significantly smaller than the straightforward $|\mathcal{H}^{(r+2)}(m+1)| = 2^{\binom{m+1}{r+2}}$ in Theorem 1. For instance, as it has been already mentioned, $n(4, 8) = 999$ which should be compared with 2^{70}.

Remark 1. Corollary 1 is implicitly used in [15] for shorter RM codes.

Next, we can state another claim that allows further reduction of the cost.

Corollary 2. *For given $e \in \mathcal{H}^{(r+2)}(m)$, let $\mathcal{H}^{(r+1)}(m)$ be partitioned into blocks (subsets) $G_i, 1 \leq i \leq s$ with the property that whenever $g \in G_i$ the enumerator $\mathcal{W}[z; e + g + \mathcal{R}(r, m)]$ is a (distinct) constant polynomial $w_i(z)$. Then the following holds:*

*(**a**) the weight enumerator of the coset $\mathcal{C}(p) = p + \mathcal{R}(r+1, m+1), p = e + f x_{m+1}$ for fixed $f \in \mathcal{H}^{(r+1)}(m)$, can be expressed by*

$$\sum_{i=1}^{s} w_i(z) \left(\sum_{g \in G_i} \mathcal{W}[z; e + g + f + \mathcal{R}(r, m)] \right).$$

*(**b**) the number of polynomial multiplications for computing the aforesaid weight enumerator equals to s, i.e. the number of distinct weight enumerators $\mathcal{W}[z; e + g + \mathcal{R}(r, m)], g \in \mathcal{H}^{(r+1)}(m)$, while that of polynomial additions is $2^{\binom{m}{r+1}} - s$.*

Proof. Rearranging the summands in (*) from Theorem 2 and putting outside of brackets the common multipliers $w_i(z)$ proves (**a**). The claim (**b**) is an immediate consequence of (**a**). □

The affine equivalence classification of $\mathcal{R}(r+2, m)/\mathcal{R}(r, m)$ enables to substantiate the usage of Corollary 2. To see this, let us recall the following definition:

Definition 5. *The subgroup $\mathcal{S}t(e)$ of $GA(m)$ that fixes $e \in \mathcal{H}^{(r+2)}(m)$, i.e., for each $A \in \mathcal{S}t(e)$ it holds: $e \circ A \in e + \mathcal{R}(r+1, m)$, is called stabilizer of e in $GA(m)$.*

For given $e \in \mathcal{H}^{(r+2)}(m)$, the stabilizer $\mathcal{S}t(e)$ partitions the cosets of the form $e + g + \mathcal{R}(r, m)$ where $g \in \mathcal{H}^{(r+1)}(m)$ into disjoint orbits. Denote this partition by $\Delta(e)$. Furthermore, Property $\mathcal{P}$ implies that the enumerator $\mathcal{W}[z; e + g + \mathcal{R}(r, m)]$ is preserved when g runs over an orbit of $\Delta(e)$. The latter permits to constitute efficiently the coarse partition $\Delta'(e) = \{G_i, 1 \leq i \leq s\}$ of $\mathcal{H}^{(r+1)}(m)$ (see, Corollary 2) by merging those orbits possessing identical weight enumerators (the latter ones being computed in advance on chosen orbit representatives).

3.2 Computing $\mathcal{W}[z;\mathcal{R}(4,9)]$

Our computational work is divided into two main phases: a pre-computing and actual computing.

The aim of pre-computing is to provide tools for efficient computation of the expression (*) in Theorem 2 given a specific e and f, and is carried out following Corollary 2 and the subsequent considerations from the previous subsection.
Let $\mathcal{E}(4,7)$ be the set of representatives of the twelve linear equivalence classes of $\mathcal{R}^*(4,7)$ given in [11]. For fixed $e \in \mathcal{E}(4,7)$, the pre-computing involves the following three tasks:

- $\mathcal{T}1$: Constitute and store the orbits of the partition $\Delta(e)$;
- $\mathcal{T}2$: Compute the weight enumerators of the cosets $e+g+\mathcal{R}(2,7)$ when g varies over a set of representatives of $\Delta(e)$'s orbits;
- $\mathcal{T}3$: Merge the orbits with identical weight enumerators to obtain the coarse partition $\Delta'(e)$, and make data arrangement permitting for given $f \in \mathcal{H}^{(3)}(7)$ to look up the identifier of a block in $\Delta'(e)$ containing f (respectively, to have direct access to the common weight enumerator).

For all $e \in \mathcal{E}(4,7)$, we present in **Table 1** of the **Appendix** the sizes of partitions $\Delta(e)$ and $\Delta'(e)$, respectively.

Remark 2. It is worth pointing out that:

- the task $\mathcal{T}1$ is efficiently performed based on the so-called "orbit algorithm" [6] using the set of generators of the stabilizer $\mathcal{S}t(e)$ provided by [11];
- the task $\mathcal{T}2$ can be carried out simultaneously for all representatives by exhaustive generation of the codewords of $\mathcal{R}(2,7)$ based on some Gray code.

Now, following the strategy described in Section 3.1, we present an algorithm for computing the weight enumerator $\mathcal{W}[z;C(p)]$ of the coset $C(p)=p+\mathcal{R}(3,8)$ where $p=e+fx_8$ for fixed $e \in \mathcal{E}(4,7)$ and a given input $f \in \mathcal{H}^{(3)}(7)$. Note that it can be implemented as a subroutine. Recall also that the common weight enumerator $w_i(z)$ corresponding to the block G_i in $\Delta'(e)$ has been already computed in the pre-computing task $\mathcal{T}2$ where $1 \le i \le |\Delta'(e)| = s(e)$.

Algorithm 1: Returning the weight enumerator $\mathcal{W}[z;C(p)]$ where $p=e+fx_8$ for fixed e and a given $f \in \mathcal{H}^{(3)}(7)$

```
U[z] := 0;
for i in [1,s(e)] do
    UU(z) := 0;
    for g in G[i] do
        j := FindBlock(g+f);
        UU(z) := UU(z) + w[j](z);
    U(z) := U(z) + w[i](z) * UU(z);
W[z; C(p)] := U(z);
```

In the actual computing, we apply formula (1) supposing that a set $\mathcal{S}$ of pairs: (representative p_i, orbit size L_i) for the $i-$th class $O_i, 1 \le i \le 999$, of the classification of $\mathcal{R}^*(4,8)$ is available. W.l.o.g., we may assume each p_i is of the form $e+f_ix_8$ for some $e \in \mathcal{E}(4,7)$ and $f_i \in \mathcal{H}^{(3)}(7)$, so the set of classes is naturally partitioned into subsets $\mathcal{O}(e)$ of cardinalities $n(e), e \in \mathcal{E}(4,7)$. (The values $n(e)$ are given in the first column of **Table 2** of the **Appendix**.) Bellow, we present an algorithm for computing the sum in formula (1) and thus $\mathcal{W}[z;\mathcal{R}(4,9)]$. (Note that we call the subroutine $\mathcal{W}[z;C(p)]$.)

Algorithm 2: Computing $\mathcal{W}[z;\mathcal{R}(4,9)]$

```
V(z) := 0;
for e ∈ E(4,7) do
    for j in [1,n(e)] do
        p := Representative(O(e)[j]);
        L := Size(O(e)[j]);
        V(z) := V(z) + L * W^2[z;C(p)];
W[z;R(4,9)] = V(z);
```

The data present in [10] contains information to form a set $\mathcal{S}'$ of kind similar to $\mathcal{S}$. However, the representatives p'_i there are of the form $e' + f'_ix_8$ where e's constitute different set of representatives of the twelve classes of $\mathcal{R}^*(4,7)$, say $\mathcal{E}'(4,7)$. For some elements of $\mathcal{E}(4,7)$ and $\mathcal{E}'(4,7)$, their linear equivalence is evident by eye inspection. For the remaining, we determined those which are linearly equivalent by computing the vectors of invariants of their duals (see, for details [7, pp. 115-117]). The matching found is represented in the rows of **Table 2** where $\overline{\mathcal{E}}(4,7)$ and $\overline{\mathcal{E}}'(4,7)$ are the sets consisting of dual forms of those in $\mathcal{E}(4,7)$ and $\mathcal{E}'(4,7)$, respectively. To find out a nonsingular (7×7) matrix $\mathbf{A}$ with property that $e' \circ \mathbf{A} \in e + \mathcal{R}(3,7)$ for thus determined pairs (e', e), we wrote a simple program in *C* which generates at random such a nonsingular square matrix and then checks the imposed condition. This technique is sufficiently efficient (due to relatively large stabilizers sizes, see, [12, **Table 2**]) and the program finished successfully its work in reasonable time. For similar technique to exploring affine equivalence of Boolean functions, we refer the reader to [14]. The obtained results are presented in the last column of **Table 2** of the **Appendix**. Finally, acting on corresponding $f'_i, 1 \leq i \leq 999$ by the resulting linear transformations (of course, ignoring the terms of degree less than 3), we obtain a type of set required by the **Algorithm 2**. The weight distribution got is presented in **Table 3** of the **Appendix**.

Remark 3. The functions FindBlock$(\cdot)$, Representative$(\cdot)$ and Size$(\cdot)$ have names that are self-explanatory when it comes to their intended purpose.

3.3 Evaluating the computational costs

Following [6] and [11], we estimate that the computational cost of task $\mathcal{T}1$ is $|\mathcal{H}^{(3)}(7)| \times \sum_{e\in\mathcal{E}(4,7)} |Sg(e)| = 2^{35} \times 26 \approx 2^{39.7}$ affine transformations where $Sg(e)$ denotes the set of generators of the stabilizer $\mathcal{S}t(e)$. The computational complexity of task $\mathcal{T}2$ is in total proportional to the product $68443 \times 2^{29} \approx 2^{45.06}$ with the first factor being the number of classes of $\mathcal{R}(4,7)/\mathcal{R}(2,7)$ and the second being the size of $\mathcal{R}(2,7)$. Task $\mathcal{T}3$ can be carried out by applying some sorting technique. In summary, the pre-computing in case $r = 2$ and $m = 7$ is efficiently performed. In addition, we note that the compressed storing of orbit and data arrangement into RAM needs at most 124 GB of memory.

In the actual computing, for every $e \in \mathcal{E}(4,7)$, **Algorithm 1** requires $|\Delta'(e)|$ multiplications and about 2^{35} additions of degree 128 polynomials with nonnegative integer coefficients. Therefore, **Algorithm 2** requires $\sum_{e \in \mathcal{E}(4,7)} n(e) \times |\Delta'(e)| = 1827252 \approx 2^{20.8}$ multiplications and about $999 \times 2^{35} \approx 2^{45}$ additions of polynomials of that kind; and 999 squarings of degree 256 polynomials and some additional operations with negligible cost, of course.

Remark 4. The straightforward application of Theorem 2 (based on the original partition $\Delta(e)$) will require about 6 times more multiplications of degree 128 polynomials than the actually executed.

Remark 5. Finally, we have two remarks concerning the implementation:

- To meet the memory limitations, we use the appropriate for that aim Delta compression and VByte encoding of the data. These techniques are important to our computer-aided solution, but the details are omitted because of their merely auxiliary role;
- We use the 256-bit CPU registers which ensures that arithmetic operations are performed efficiently and eliminates the need to further estimate the number of processor operations.

4 Conclusion

In this article, thanks to recent advances in the classification of Boolean functions [5],[12] and the utilization of modern high-performance computers, a solution to the problem at hand is obtained. However, we should admit that it may not be doable to push this line of research much further due to the way in which the computational burden increases with code length.

Acknowledgments

We would like to thank the reviewers for their valuable suggestions and comments which significantly improve this paper. We are also grateful to Vladimir Tonchev for pointing out the problem and his stimulating discussions, as well as to Peter Boyvalenkov for his interest and support during the course of this research.

This work was supported, in part, by the Ministry of Education and Science of Bulgaria under the Grant No. DO1-325/01.12.2023 "National Centre for High-performance and Distributed Computing" (NCHDC). The authors acknowledge the provided access to the e-infrastructure of NCHDC.

Appendix

Table 1. Sizes of partitions $\varDelta(e)$ and $\varDelta'(e)$

$e \in \mathcal{E}(4,7)$: ANF's according to ([11])	$\lvert\varDelta(e)\rvert$	$\lvert\varDelta'(e)\rvert$
0	12	12
4567	63	52
1235+1345+1356+1456+2346+2356+2456	130	112
2367+4567	289	182
1237+4567	480	306
1257+1367+4567	730	395
1237+1247+1357+2367+4567	204	157
1236+1257+1345+1467+2347+2456+3567	1098	675
1236+1356+1567+2357+2467+2567+3456	1340	811
1367+2345+2356+3456+4567	6449	2170
1234+1237+1267+1567+2345+3456+4567	23988	3377
1236+1367+1567+2345+3456+3457+3467	33660	4636

Table 2. The matching between $\mathcal{E}'(4,7)$ and $\mathcal{E}(4,7)$

Distribution of $n(e)$	$\overline{\mathcal{E}}'(4,7)$	$\overline{\mathcal{E}}(4,7)$	Linear transition transformation
3	0	0	[1000000 0100000 0010000 0001000 0000100 0000010 0000001]
2	123	123	[1000000 0100000 0010000 0001000 0000100 0000010 0000001]
21	127+136+145	137+147+157+237+247+267+467	[0011001 0011110 0100110 1011000 1111010 1001100 0001100]
15	125+134	123+145	[1000000 0100000 0001000 0000100 0010000 0000010 0000001]
89	126+345	123+456	[1000000 0100000 0001000 0000100 0000010 0010000 0000001]
56	126+135+234	123+245+346	[0100000 0010000 0001000 0000010 0000100 1000000 0000001]
10	135+146+235+236+245	123+145+246+356+456	[1000000 0000010 0001000 0010000 0000100 0100000 0000001]
7	127+136+145+234	124+137+156+235+267+346+457	[0110001 1011001 0110011 0111010 1100101 0010111 1001011]
502	125+134+135+167+247+357	127+134+135+146+234+247+457	[0001000 0010000 0000001 0000100 0100000 0000010 1100110]
1	123+247+356	123+127+147+167+245	[0010000 0110011 1010000 0001110 0000001 0010011 0000100]
1	147+156+237+246+345	123+127+167+234+345+456+567	[0101010 1001010 1001001 1111111 0011000 0100010 1001011]
292	127+146+236+345	125+126+127+167+234+245+457	[0100111 0001110 0110110 1011000 0000010 0000100 0010110]

Table 3. Weight Distribution of the [512,256,32] Reed-Muller code

Weight	Number of codewords
0 512	1
32 480	52955952
48 464	919315326720
56 456	271767121346560
60 452	860689275027456
64 448	89163020044002040
68 444	1777323352931696640
72 440	64959328938397057024
76 436	2094952122987829002240
80 432	86129855718211879936768
84 428	3718387228743293604986880
88 424	216407674400647746861465600
92 420	15958945395035022932054114304
96 416	1570964763114053055495174389136
100 412	207755244457303752035637154283520
104 408	34164336816436357675455725024378880
108 404	5992987676360073735151889707696128000
112 400	983217921810034263357552475089021004288
116 396	140881159168600922710983130625456163782656
120 392	17178463264607761296016540993629780705771520
124 388	1770270551281316280504947079180771901717872640
128 384	154198773988541804525321284585063483246993999900
132 380	11380437366712812474455950864177326068447989202944
136 376	713793445298874211607839796879716106185715280216064
140 372	38161660034401312989486264769054124765959796671119360
144 368	1744077996406613042017016863461234839306732612077058560
148 364	68320936493023612641136928149296775084064365913214812160
152 360	2299744204800465802453316637595783829108912802028206751744
156 356	66674424868716978552789375387240003239187186349775851094016
160 352	1668559700964160587350805664583122924498928358151715733007408
164 348	36117082274027891545154187373048131661136552390031364702863360
168 344	677483598989547107793615101247739514269621184741356041461104640
172 340	11032441933713096201663286389373184730113421621201515757397082112
176 336	156225095497619813307679231937780861426835567156776476525084177664
180 332	1926667532217097161576702991776654344250440175688196887457279508480
184 328	20723534026876536792281002394151796205045793736436788802938336133120
188 324	194671442741837852939975553363771856234841259238404365556287065292800
192 320	1599044990181340998819270766161596605692512085057170791477694075282632
196 316	11498415685246302189888474222781442491860129957714864173250891967627264
200 312	72459467570743603819378812718772497540870770484626494838959726267809792
204 308	400549932263936554220342987258224499780564121712827465674395223861493760
208 304	1944071611978423909059426198144849863064608675044397429548995177751732480
212 300	8291211853278378544436157221213736835450108801042695204524353086973542400
216 296	31095502600701130763682713427899390240950550846409105550583369693522427904
220 292	102622652435510219354959437959897900434480615845926142166854426192158654464
224 288	298206281302110726623000750445450132512881810629607123478473554095237810960
228 284	763396919631666688676755106996803883003881847438728311891109384630797598720
232 280	1722452776176219896357452486934573175804665343735169479919087899582551687168
236 276	3426750460257305904470547641506642175867699465315478403354123631366508642304
240 272	6013163599489683999312799935491777179772724247998877953378442920501417933824
244 268	9309551320248854051332692772889245412495562988894547412532818045057116405760
248 264	12718986044129514620716674156341900030463015021774940408815989741288144568320
252 260	15336997499945305387056357527918950456934399969250231086077675815418680311808
256	16324199909251682000435577287934368523097397692548071777837483832108326674502

References

1. C. Carlet, *Boolean Functions for Cryptography and Coding Theory*, Cambridge University Press, Cambridge, 2021.
2. C. Carlet and P. Solé, "The weight spectrum of two families of Reed-Muller codes", Discrete Mathematics, **346(10)**, 113568, 2023.
3. P. Charpin, "Open problems on cyclic codes", in *Handbook of Coding Theory*, vol. 1, V. S. Pless, C. W. Huffman, editors, R. A. Brualdi, assistant editor, Elsevier, 963-1063, 1998.
4. Th. W. Cusick and P. Stănică, *Cryptographic Boolean functions and Applications*, Academic Press, Amsterdam,..., Tokyo, 2009.
5. V. Gillot and Ph. Langevin, "Classification of some cosets of the Reed-Muller code", Cryptogr. Commun. (2023), available at https://doi.org/10.1007/s12095-023-00652-4.
6. A. Hulpke, "Computing with group orbits", available at https://www.math.colostate.edu/
7. X. -D. Hou, "$GL(m, 2)$ acting on $\mathcal{R}(r, m)/\mathcal{R}(r-1, m)$", Discrete Mathematics, **149**, 99-122, 1996.
8. T. Kasami and N. Tokura, "On the weight structure of Reed-Muller codes", IEEE Trans. Info. Theory, 16 , 752-759, 1970.
9. T. Kasami, N. Tokura, S. Azumi, "On the weight enumeration of weights less than 2.5d of Reed-Muller codes", Information and Control, 30, 380-395, 1976.
10. Ph. Langevin, "Classification of Boolean quartic forms in eight variables", available at https://langevin.univ-tln.fr/project/quartics/quartics.html, 2007.
11. Ph. Langevin, "Classification of $RM(4, 7)/RM(2, 7)$", available at https://langevin.univ-tln.fr/project/rm742/rm742.html, 2012.
12. Ph. Langevin and G. Leander, "Classification of Boolean quartic forms in eight variables", in *Boolean Functions in Cryptology and Information Security*, B. Preneel and O. A. Logachev (Eds.), IOS Press, 139-147, 2008.
13. F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*, North-Holland Publishing Company, Amsterdam, New York, Oxford, 1977.
14. Q. Meng et al., "Analysis of affinely equivalent Boolean functions", Science in China Series F: Information Sciences., vol. 50, no. 3, 299-306, 2007.
15. D. V. Sarwate, *Weight Enumeration of Reed-Muller Codes and Cosets*, Ph.D., Dep. Elec. Eng., Princeton Univ., Princeton, N.J., Sept. 1973, Advisors: E. R. Berlekamp and J. D. Ullman.
16. N. J. A. Sloane, "On-line Encyclopedia of Integer Sequences". available at https://oeis.org/wiki/List of weight distributions
17. N. J. A. Sloane, E. R. Berlekamp, "Weight enumerator for second-order Reed-Muller codes", IEEE Trans. Info. Theory, 16, 745-751, 1970.
18. Ts. Sugita, T. Kasami, and T. Fujiwara, "The weight distribution of the third-order Reed-Muller code of length 512", IEEE Trans. Info. Theory, 42, No. 5, 1622-1625, 1996.
19. M. Sugino, Y. Tokura and T. Kasami, "Weight distribution of (128,64) Reed-Muller Code", IEEE Trans. Info. Theory, 17, 627-628, 1971.
20. H. C. A. van Tilborg, "Weights in the third-order Reed-Muller codes", JPL Technical Report 32-1526, vol.IV, 86-92, 1971.