Secure Majority Voting for Multi-Rater Labelling

1st Taejeong Kim
Dept. of Electronic Engineering
Hanyang University
Seoul, Korea
birdy0212@hanyang.ac.kr

2nd Dong-Joon Shin
Dept. of Electronic Engineering
Hanyang University
Seoul, Korea
djshin@hanyang.ac.kr

***Abstract*—Majority voting over binary labels is a fundamental operation in many machine learning pipelines, such as weak supervision, multi-rater annotation, and 1-bit compressed optimization. In these settings, individual labels or bits are often sensitive, while only the final majority voting outcome needs to be revealed. This paper presents a compact description of how to securely compute the majority vote by combining 1-bit multi-party computation (MPC) based on binary Reed–Muller (RM) codes with efficient majority circuits. The RM-based secret sharing keeps all inputs, intermediate values, and outputs in the 1-bit domain. The majority function is realized as a Boolean circuit using 3-input majority gates, interpreted over the shares as additions and multiplications in $\mathbb{F}_2$. We outline the protocol, discuss its security and communication complexity, and highlight why the RM–MPC framework is particularly well-suited for secure majority voting on binary data.**

***Index Terms*—MPC, Multi-Rater Labelling, Secret Sharing, RM codes**

I. Introduction

Binary decisions of the form *yes/no* or *positive/negative* appear in many modern AI systems. Examples include whether a medical image shows a disease, whether a piece of content violates a policy, or whether a classifier predicts a sample as positive or negative. In practice, such decisions are often made by multiple agents—experts, models, or heuristic rules—and combined by majority voting. In crowd labeling, several annotators vote on a sample and the final label is determined by majority. In weak supervision, a set of noisy labeling functions output $0/1$ labels and a label model aggregates them and decide by majority vote. In multi-rater medical annotation, two or more clinicians each provide a binary diagnosis, and a consensus label is determinded by majority vote.

While the majority outcome is typically the only quantity required by downstream training or inference, the individual $0/1$ labels are sensitive. In medical or financial domains, revealing which expert or institution voted "1" on which sample may leak private diagnostic criteria, internal policies, or even legal exposure. In learning systems, revealing per-source labels may expose proprietary models or heuristic rules. Consequently, there is a strong motivation to compute the majority vote in a privacy-preserving way: only the final aggregate bit should be revealed, and all individual inputs must remain secret.

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT)(No. RS-2024-00409492).

Generic MPC or homomorphic encryption can be used to implement such functionality, but they often operate over larger fields and require arithmetic circuits that are unnecessarily heavy even when all inputs are binary. To accommodate the cases mentioned above, we consider the following setting This paper focuses on a setting:

- the inputs are bits $x_1, \ldots, x_n \in \{0, 1\}$;
- the desired output is the majority bit

$$\mathrm{Maj}(x_1, \ldots, x_n) = \mathbf{1}\Big[\sum_{i=1}^{n} x_i \geq T\Big],$$

  the indicator function that equals 1 if $\sum_{i=1}^{n} x_i \geq T$ and 0 otherwise; and
- we wish to keep the 1-bit representation throughout the computation.

For this special but important case, 1-bit MPC based on binary Reed–Muller(RM) codes and majority circuits provides a natural and efficient solution.

II. Preliminaries

A. RM Codes and 1-Bit Secret Sharing

Let $\mathbb{F}_2[x_1, \ldots, x_m]$ denote the ring of multivariate polynomials over the field $\mathbb{F}_2$. For a polynomial f and point $u \in \mathbb{F}_2^m$, $f(u)$ is the evaluation of f at u. The binary RM code of order r and length 2^m is defined as

$$\mathrm{RM}(r, m) \triangleq \{(f(u))_{u \in \mathbb{F}_2^m} : f \in \mathbb{F}_2[x_1, \ldots, x_m],\ \deg(f) \leq r\}.$$

In the 1-bit secret sharing scheme that we consider, the number of parties n is set to $2^m - 1$, and a code $\mathrm{RM}(r, m)$ with $r = \lfloor (m-1)/2 \rfloor$ is used. A binary secret $s \in \{0, 1\}$ is embedded as the first coordinate of a random codeword $(c_0, c_1, \ldots, c_n) \in \mathrm{RM}(r, m)$ with $c_0 = s$. Each party P_i,$1 \leq i \leq n$, receives a single bit $c_{\pi(i)}$ for a random permutation π on $\{1, 2, ..., n\}$; these bits serve as the shares of s [1].

This construction has two key properties. First, each share is a single bit, and the linear operation XOR on secrets correspond to the bit-wise XOR of the shares, since $\mathrm{RM}(r, m)$ is a linear code. Second, the algebraic structure of RM codes allows efficient handling of Boolean circuits when combined with standard MPC techniques.

B. 1-Bit MPC over $\mathbb{F}_2$

To evaluate Boolean circuits securely, we need to support addition and multiplication of shared bits. Let $[s]^{\mathrm{RM}} = (s_1^{\mathrm{RM}}, \ldots, s_n^{\mathrm{RM}})$ denote the RM shares of a secret $s \in \{0,1\}$, and let $[s]^{n\text{-of-}n}$ denote an n-out-of-n sharing where $s = \sum_{i=1}^{n} s_i^{n\text{-of-}n}$ over $\mathbb{F}_2$.

a) Addition: The notation $\oplus$ denotes XOR gate. Given RM shares $[a]^{\mathrm{RM}}$ and $[b]^{\mathrm{RM}}$, each party P_i locally computes $c_i^{\mathrm{RM}} = a_i^{\mathrm{RM}} \oplus b_i^{\mathrm{RM}}$. The resulting vector c_i^{RM} is again an RM codeword sharing the secret $c = a \oplus b$ [1].

b) Multiplication: For multiplication, the component-wise product $a_i^{\mathrm{RM}} b_i^{\mathrm{RM}}$ corresponds to the evaluation of the product polynomial $f \cdot g$, whose degree may exceed r and thus leave the code $\mathrm{RM}(r,m)$. To bring the result back into the code, a standard trick is used: for each multiplication gate, a random mask $s \in \{0,1\}$ is pre-shared in both RM form $[s]^{\mathrm{RM}}$ and n-out-of-n form $[s]^{n\text{-of-}n}$ in the offline phase. During the online phase, parties locally compute

$$d_i^{n\text{-of-}n} = a_i^{\mathrm{RM}} b_i^{\mathrm{RM}} \oplus s_i^{n\text{-of-}n}$$

and send $d_i^{n\text{-of-}n}$ to a designated aggregator. The aggregator sums all $d_i^{n\text{-of-}n}$ to obtain

$$d = \bigoplus_{i=1}^{n} d_i^{n\text{-of-}n} = a \cdot b \oplus s,$$

and broadcasts d to all parties. Each party then sets

$$c_i^{\mathrm{RM}} = s_i^{\mathrm{RM}} \oplus d^{\mathrm{RM}},$$

where d^{RM} is the RM share of the constant d (all-zero codeword if $d = 0$, all-one codeword if $d = 1$). The resulting vector $(c_i^{\mathrm{RM}})_i$ shares the product bit $a \cdot b$ and still lies in $\mathrm{RM}(r,m)$, so it can be used as input to further gates [1].

By composing these addition and multiplication gadgets, any Boolean circuit can be evaluated securely on 1-bit shares. Note that the precomputation of random masks can be done offline, independent of the actual inputs.

In summary, a 1-bit MPC protocol over $\mathbb{F}_2$ based on RM codes represents each secret bit as an RM codeword share and implements addition and multiplication gates by local operations on these shares. Addition corresponds to component-wise XOR, while multiplication uses preprocessed random masks shared both in RM form and n-out-of-n form so that the (potentially high-degree) product of codewords can be re-randomized and brought back into $\mathrm{RM}(r,m)$ with one round of communication. By composing these gadgets, any Boolean circuit can be securely evaluated on 1-bit RM shares, with the preprocessing of random masks being independent of the actual inputs [1].

C. Majority Voting Circuit

The target function in this work is the majority (or more generally, threshold) function. For n binary inputs $x_1, \ldots, x_n$, the T-out-of-n threshold function outputs 1 if at least T of its inputs are 1, and 0 otherwise. The majority vote corresponds to $T = \lceil n/2 \rceil$.

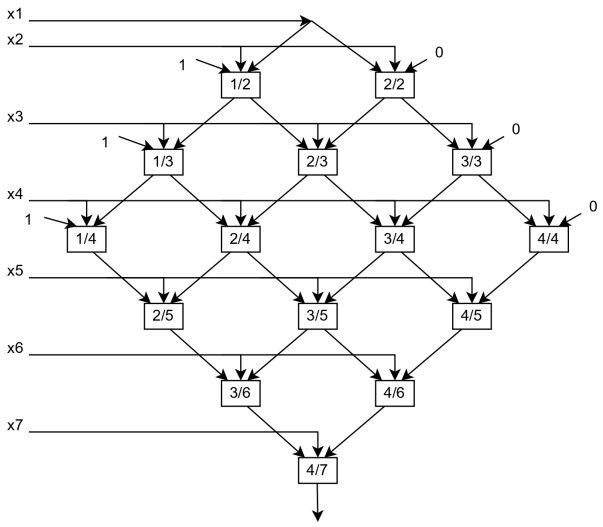

Fig. 1. $(4/7)$ majority circuit

A convenient building block is the 3-input majority gate defined as

$$\mathrm{maj}(x_1, x_2, x_3) = \begin{cases} 1, & \text{if at least two inputs are 1,} \\ 0, & \text{otherwise.} \end{cases}$$

This gate can be implemented by three AND gates and two XOR gates:

$$\mathrm{maj}(x_1, x_2, x_3) = x_1 x_2 \oplus x_2 x_3 \oplus x_3 x_1. \tag{1}$$

Interpreting XOR as addition and AND as multiplication over $\mathbb{F}_2$, this representation is directly compatible with the 1-bit MPC primitives.

For general n, threshold functions can be realized by connecting 3-input majority gates in a diamond-shaped network. As shown in Figure 1, each gate in the network is associated with an (a/b) function: it outputs 1 if at least a of its b designated inputs (original input bits or outputs of upper gates) are 1 and 0 otherwise. By layering these gates appropriately, one can construct an (a/n) circuit for arbitrary a and n, with the number of 3-input majority gates bounded by $a(n-a+1)-1$. The depth of the network grows linearly in n, but gates in the same row are independent and can be evaluated in parallel[2].

In our setting, we instantiate such a circuit for the desired threshold T. All inputs and gate outputs are represented by RM shares, and each 3-input gate is evaluated using one secure multiplication gadget per AND gate in its decomposition as is Equation(1).

III. Secure Majority Voting Protocol

We now describe the protocol that securely computes the majority bit of n private inputs $x_1, \ldots, x_n \in \{0,1\}$.

A. Setting and Threat Model

There are n parties $P_1, \ldots, P_n$. where each P_i holds a private bit x_i. The goal is to compute $y = \mathrm{Maj}(x_1, \ldots, x_n)$ such that:

- y is revealed to an agreed party (e.g., all parties or a server),
- no coalition of fewer than a specified number of parties learns any additional information about the individual inputs beyond what is implied by y.

We assume a semi-honest adversary model: corrupt parties follow the protocol, but try to infer extra information from the messages they see. The RM-based sharing is instantiated with parameters such that individual shares are statistically independent of the secret, up to the chosen privacy threshold.

B. Protocol Description

We now describe the secure majority voting protocol based on 1-bit RM–MPC. There are n parties $P_1, \ldots, P_n$, where each party P_i holds a private bit $x_i \in \{0, 1\}$. The goal is to compute the majority bit $y = \mathrm{Maj}(x_1, \ldots, x_n)$ without revealing any individual x_i.

In the **preprocessing (offline) phase**, the parties first fix a majority circuit C for n inputs, built from 3-input majority gates. Let B denote the number of AND gates in this circuit. For each AND gate index $j \in \{1, \ldots, B\}$, a random mask bit s_j is sampled and then shared in two ways: as RM shares $[s_j]^{\mathrm{RM}}$ and as n-out-of-n shares $[s_j]^{n\text{-of-}n}$. These shares are distributed so that each user P_i receives one RM share $s_{j,i}^{\mathrm{RM}}$ and one n-out-of-n share $s_{j,i}^{n\text{-of-}n}$ for every AND gate j. This preprocessing depends only on the circuit structure, not on the actual inputs.

In the **online phase**, each user P_i first secret-shares their input bit x_i by encoding it as RM shares $[x_i]^{\mathrm{RM}}$ and distributing the shares so that every user holds one share of each input. With these shares in place, all XOR gates of the circuit C are evaluated locally: since XOR corresponds to addition in $\mathbb{F}_2$, each party simply adds the corresponding RM shares to obtain the output shares for every XOR gate.

For each AND gate g in the circuit C, a standard masked multiplication is performed using the pre-shared masks. Suppose that a some party P_i has the RM shares a_i^{RM} and b_i^{RM} of the two inputs to the AND gate of g. Let $s_{j,i}^{\mathrm{RM}}$ and $s_{j,i}^{n\text{-of-}n}$ be the RM and n-out-of-n shares of the mask s_j associated with this gate. Then P_i computes

$$d_i^{n\text{-of-}n} = a_i^{\mathrm{RM}} b_i^{\mathrm{RM}} \oplus s_{j,i}^{n\text{-of-}n}$$

and sends $d_i^{n\text{-of-}n}$ to the designated aggregator. The aggregator XORs all received values to obtain

$$d = \bigoplus_i d_i^{n\text{-of-}n} = a \cdot b \oplus s_j,$$

and broadcasts this single bit d to all parties. Each party P_i then reconstructs the output share of the gate by setting

$$c_i^{\mathrm{RM}} = s_{j,i}^{\mathrm{RM}} \oplus d^{\mathrm{RM}},$$

where d^{RM} is the RM share of the public bit d (the all-zero codeword if $d = 0$, and the all-one codeword if $d = 1$). Repeating this procedure for every AND gate yields RM shares for all internal nodes of the circuit.

After all gates in C have been evaluated, each user P_i holds an RM share y_i^{RM} of the circuit output, which encodes the majority bit y. In the final reconstruction step, the parties send their output shares y_i^{RM} to the designated decoder (or exchange them among all users), and the decoder applies the RM decoding algorithm to recover the plaintext output y. This bit is then delivered to the intended recipients.

In summary, all operations in the protocol are performed on 1-bit shares; no large moduli or heavy ciphertext arithmetic are involved. XOR gates require only local computation on the shares, while each AND gate incurs a single round of communication. The final decoding step can be implemented either centrally by a server or in a decentralized manner by the users themselves.

IV. Security and Complexity

A. Security

We briefly summarize the privacy guarantee of the RM-based secret sharing. A secret bit $s \in \{0, 1\}$ is embedded as the first coordinate of a random codeword $c = (c_0, \ldots, c_n) \in \mathrm{RM}(r, m)$ with $c_0 = s$, and each party P_i receives its share c_i. For any subset $S \subseteq \{1, \ldots, n\}$, let

$$\mathsf{View}_s(S) = (c_i)_{i \in S}$$

denote the shares observed by S when the secret is s. Privacy means that coalitions of size at most t cannot distinguish $s = 0$ from $s = 1$:

$$\Delta\big(\mathsf{View}_0(S), \mathsf{View}_1(S)\big) \le \varepsilon_n \quad \text{for all } S,\ |S| \le t, \tag{2}$$

where ε_n is negligible in n. Here, $\Delta(\cdot, \cdot)$ denotes the statistical distance between two distributions, i.e., for random variables X, Y over a common finite domain $\mathcal{X}$,

$$\Delta(X, Y) \triangleq \tfrac{1}{2} \sum_{x \in \mathcal{X}} \big|\Pr[X = x] - \Pr[Y = x]\big|.$$

For suitable parameters (r, m), the structure of binary RM codes and their duals implies the existence of a threshold t arbitrarily close to $n/2$ (see, e.g.,[1]). Equivalently, for any fixed $\delta > 0$ and

$$t = \Big(\frac{1}{2} - \delta\Big) n,$$

Equation (2) holds with $\varepsilon_n \le 2^{-\Omega(n)}$. Thus, any coalition controlling strictly less than 50% of the shares learns essentially no information about the secret bit s [2],[3].

Since 1-bit RM MPC protocol only reveals the final majority $y = \mathrm{Maj}(x_1, \ldots, x_n)$ in plaintext and keeps all intermediate values secret-shared and masked, any adversary corrupting fewer than 50% of parties learns nothing about the honest users' inputs beyond what is implied by the output y itself.

B. Communication and Round Complexities

Let B denote the number of 3-input majority gates in the chosen threshold circuit. Each such gate uses three AND gates and two XOR gates, so the total number of AND gates is $3M$. For each AND gate, every user sends and receives one bit, resulting in a per-user communication cost proportional to $6M$ bits, plus a constant number of bits for output reconstruction. XOR gates are free in terms of communication because they are locally evaluated.

The circuit depth grows linearly with n as shown in Figure 1, and gates in each layer can be evaluated in parallel. Consequently, the total number of MPC rounds is linear in n plus a small constant for reconstruction. While optimizing the majority network to minimize the number of AND gates is an interesting direction, even straightforward constructions in this work already yield a practical secure majority primitive for moderate numbers of users.

V. Conclusion

This paper extracted and highlighted the core mechanism of combining 1-bit RM MPC with the majority circuits to securely compute majority vote on binary inputs. The RM-based sharing keeps all values in the 1-bit domain, while the majority circuit provides a structured way to realize threshold functions using only AND and XOR gates. Together, they form an efficient building block for privacy-preserving majority decisions in applications such as weak supervision, crowd labeling, and multi-rater annotation. Future work includes optimizing majority circuits for lower multiplicative complexity and integrating this primitive into concrete end-to-end AI pipelines.

References

[1] Applebaum, B. and Kachlon, E. Stochastic Secret Sharing with 1-Bit Shares and Applications to MPC. In Annual International Cryptology Conference, pp.4-13. Springer, 2024.

[2] Amarel, Saul, G. Cooke, and Robert O. Winder. Majority gate networks. In IEEE Transactions on Electronic Computers 1, 2006.

[3] Kudekar, S., Kumar, S., Mondelli, M., Pfister, H. D., and Urbanke, R. Reed-muller codes achieve capacity on erasure channels. In Proceedings of the forty- eighth annual ACM symposium on Theory of Computing, pp. 658–669, 2016.