An Algebraic Method for Constructing Bases in Binary Linear Codes for Information Dispersal Algorithms

Oscar Casimiro-Muñoz , Ricardo Marcelín-Jiménez , Rubén Vázquez-Medina * , Leonardo Palacios-Luengas *

Posted Date: 14 February 2026

doi: 10.20944/preprints202602.1166.v1

Keywords: linear codes; generalized Hamming weight, information dispersal algorithm; binary linear code; polynomial ring

Article

An Algebraic Method for Constructing Bases in Binary Linear Codes for Information Dispersal Algorithms

Oscar Casimiro–Muñoz [1], Ricardo Marcelín–Jiménez [2], Rubén Vázquez–Medina [3,*] and Leonardo Palacios–Luengas [2,*]

[1] Department of Mathematics, Autonomous Metropolitan University (UAM), Iztapalapa, Mexico City 09340, Mexico
[2] Department of Electrical Engineering, Autonomous Metropolitan University (UAM), Iztapalapa, Mexico City 09340, Mexico
[3] Instituto Politécnico Nacional, Centro de Investigación en Ciencia Aplicada y Tecnología Avanzada Unidad Querétaro, 76090, Querétaro, Mexico
* Correspondence: ruvazquez@ipn.mx (R.V.-M.); lpl@xanum.uam.mx (L.P.-L.)

Abstract

The algebraic analysis of linear code parameters reveals deep connections with cryptographic constructions, including the information dispersal algorithms (IDAs) and secret-sharing schemes. In this work, we propose an algebraic method for constructing bases of binary linear codes from subsets of codewords selected according to their generalized Hamming weights (GHWs). The approach employs a degree-compatible monomial ordering on the polynomial ring $\mathbb{F}_2[x_1, \ldots, x_n]$ and imposes the conditions $d_1(C) = 1$ and $d_k(C) = n$. Under these assumptions, we prove the existence of a generator matrix containing an invertible $k \times k$ submatrix, which guarantees correct information reconstruction. This structural property enables the direct application of binary linear codes to information dispersal and recovery mechanisms without the need for larger finite fields. We validate the proposed framework through algebraic proofs and an explicit example illustrating both the dispersal and recovery procedures. These results provide a theoretical foundation for the design of information dispersal schemes relying exclusively on binary linear codes.

Keywords: linear codes; generalized Hamming weight, information dispersal algorithm; binary linear code; polynomial ring

1. Introduction

The study of linear codes provides a mathematical foundation for secure communication, error detection and correction, and information protection. Furthermore, it underpins modern cryptographic and distributed systems, such as distributed storage, secure multiparty computation, secret-sharing schemes, and information dispersal algorithms (IDAs), along with their associated information recovery algorithms (IRAs) [1,2,8,15,16]. IDAs, introduced by Rabin [14], disperse data into n pieces, allowing reconstruction from any $t < n$ pieces to ensure reliability and fault tolerance. Rabin's construction employs matrices over algebraic structures such as polynomial rings, Euclidean rings, or finite fields [10], for example, Cauchy or Vandermonde matrices with invertible submatrices that enable reconstruction[19].

Binary linear codes are seldom used in IDAs due to generator matrices lacking invertible $k \times k$ submatrices, unlike those over $\mathbb{R}$ or $\mathbb{C}$. However, parameters like minimum distance and generalized Hamming weights (GHWs) can indicate such properties. Binary codes, being fundamental, offer rigid structures ideal for studying GHWs, bases, and invertible submatrices. We focus on algebraic conditions for binary codes to support IDAs. We show that codes with $d_1(C) = 1$ and $d_k(C) = n$ enable IDAs via an invertible $k \times k$ submatrix in the generator matrix, ensuring Rabin's IDA validity. The method uses a degree compatible monomial order on $\mathbb{F}_2[x_1, \ldots, x_n]$ to select codewords matching GHWs, forming a basis for binary IDAs without larger-field overhead. Sections 4.2 and 4.3 of [3]

 doi:10.20944/preprints202602.1166.v1

motivated us to consider dispersal matrices given by binary linear codewords with special characteristics, as well as those in [2], which allowed us to study the secret-sharing schemes and, therefore, the information dispersal algorithms in a more current way.

The paper is organized as follows. Section 2 reviews the necessary background on linear codes, GHWs, monomial orders, and IDAs. Section 3 presents the main algebraic construction, including the selection of a basis guided by GHWs and the definition of the IDA and IRA procedures. Section 4 illustrates the proposed framework through an example of a binary linear code that satisfies the required conditions. Section 5 presents the conclusions and outlines directions for future research. Finally, Appendix 5 details the method proposed by Johnsen and Verdure [6] for calculating the GHWs of a linear code.

1.1. Literature Review

This work intersects linear coding theory, algebraic methods, and IDAs. The relevant literature is summarized as follows:

1. The algebraic foundations of linear codes, including generator matrices, parity-check matrices, and structural parameters, are classical and well established; see, for instance, [12,15,16]. These works provide the basic framework for understanding linear codes as vector spaces over finite fields and for analyzing their fundamental parameters.
2. Generalized Hamming weights (GHWs), introduced by Wei [18], extend the notion of minimum distance and have become central invariants in the structural analysis of linear codes. Subsequent developments connected GHWs with algebraic and combinatorial objects, particularly through the study of matroids and Stanley-Reisner ideals. In this direction, Johnsen and Verdure [6,7] established a deep relationship between GHWs and graded free resolutions, while García-Marco et al. [5] further explored these ideas in the binary case. These results highlight the role of GHWs as tools for understanding the internal structure of linear codes beyond error-correcting capabilities.
3. From the algebraic point of view, polynomial rings and monomial orders play an important role in the study of linear codes and their associated ideals; see, e.g., [11]. Degree-compatible monomial orders provide a natural way to organize monomials according to their weight, making them suitable for linking algebraic structures with combinatorial properties of codes.
4. Information dispersal algorithms (IDA) were introduced by Rabin [1,2,14] as a method for distributing information among multiple participants in a fault-tolerant manner. Classical constructions of IDA usually use matrices from large finite fields, such as Vandermonde or Cauchy matrices, to ensure the existence of invertible submatrices [19]. While these approaches are effective, they inherently depend on non-binary fields.

1.2. Contributions

This work presents an algebraic method for constructing bases of binary linear codes with structural properties suitable for information dispersal. The contributions of this paper can be summarized as follows.

- An algebraic method is introduced for selecting a basis of an $[n, k]$ binary linear code by combining GHWs with a degree-compatible monomial order on the polynomial ring $\mathbb{F}_2[x_1, \ldots, x_n]$. The resulting basis reflects the hierarchy of supports determined by the GHWs of the code.
- Under the structural assumptions $d_1(C) = 1$ and $d_k(C) = n$, we prove that the basis obtained through this construction yields a generator matrix containing at least one invertible $k \times k$ submatrix. This result establishes a direct link between GHWs and the existence of invertible substructures in binary generator matrices.
- We provide a systematic procedure for identifying an invertible $k \times k$ submatrix from the generator matrix associated with the constructed basis. The procedure is derived from the algebraic properties of the selected codewords and does not rely on probabilistic or numerical arguments.

 doi:10.20944/preprints202602.1166.v1

- Based on the above structural results, we formalize an information dispersal and reconstruction scheme defined entirely over the binary field. The correctness of the reconstruction follows from the algebraic properties of the constructed basis and the imposed GHWs conditions.

This work highlights new ideas on GHWs that depart from the traditional perspective of error or security analysis. Instead, they are used as structural invariants that provide information about the possible supports of subcodes and, consequently, about the existence of codewords with controlled support properties. These features will be exploited in Section 2 to guide the selection of a suitable basis for the code.

2. Preliminaries

This section establishes the fundamentals of linear coding theory, specifically generator matrices, parity-check matrices, and Generalized Hamming Weights (GHWs), adopting the standard definitions and notation found in [12,15,16,18]. Furthermore, we detail the use of polynomial rings and monomial orders as discussed in [11]. In this context, these algebraic concepts serve as a structural tool to impose a strict hierarchy on codeword coordinates, facilitating the algebraic analysis of supports and GHWs. Finally, we formally define the Information Dispersal Algorithm (IDA), based on the framework presented in [1].

2.1. Linear Codes and Information Dispersal Algorithm

Let $q = p^r$ be a positive integer, where p is a prime number and r is a positive integer. The finite field with q elements is denoted by $\mathbb{F}_q$. Let n, k, d be positive integers such that $n \geq k$ and $d \geq 1$. An $[n, k, d]$ *linear code* C is defined as a subspace of the vector space $\mathbb{F}_q^n$.

A matrix $G \in \mathrm{Mat}_{k\times n}(\mathbb{F}_q)$ is called a *generator matrix* of C if its rows form a basis of C. Similarly, a matrix $H \in \mathrm{Mat}_{(n-k)\times n}(\mathbb{F}_q)$ is said to be a *parity-check matrix* of C if its null space coincides with C, that is, $xH^T = 0 \quad$ for all $x \in C$, where 0 denotes the zero vector in $\mathbb{F}_q^{n-k}$. The code C can be generated by multiplying any vector $x \in \mathbb{F}_q^k$ by the generator matrix G, that is, by computing the product xG.

Given two nonzero vectors $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$ in $\mathbb{F}_q^n$, the *support* of x is defined as $\mathrm{supp}(x) = \{i \mid x_i \neq 0\}$. The *Hamming distance* between x and y is given by

$$d(x, y) = |\{i \mid x_i \neq y_i\}|.$$

The *Hamming weight* of a vector x is defined as $w(x) = d(x, \bar{0})$, where $\bar{0}$ denotes the zero vector. The *minimum distance* of a linear code C is then defined as

$$d = \min\{w(x) \mid x \in C,\ x \neq \bar{0}\}.$$

When the minimum distance of a linear code C is known, the code is described as an $[n, k, d]$ linear code. The elements of C are called *codewords*, and the parameters n, k, and d are referred to as the *basic parameters* of the code, representing:

1. n, the *length* of the codewords;
2. k, the *dimension* of C as a vector space over $\mathbb{F}_q$;
3. d, the *minimum distance* of C.

The *support of a subset* $D \subseteq C$ is defined as

$$\mathrm{supp}(D) = \{i \mid \exists\, c \in D \text{ such that } c_i \neq 0\}.$$

In the case of a binary linear code, the support of a subset $D \subseteq C$ coincides with the set of coordinate positions for which at least one element of D has a nonzero entry.

Definition 2.1 ([5]). *The h-th* generalized Hamming weight *of a linear code C is defined as*

$$d_h(C) = \min\{|\mathrm{supp}(E)| \mid E \in D_h(C)\},$$

where $D_h(C)$ denotes the collection of all h–dimensional linear subspaces of C, for $h \in \{1, \ldots, k\}$.

 doi:10.20944/preprints202602.1166.v1

It is worth noting that when $h = 1$, the generalized Hamming weight coincides with the minimum distance of the code, that is, $d_1(C) = d$. Hence, Definition 2.1 generalizes the classical notion of minimum distance.

The following theorem establishes fundamental properties of the generalized Hamming weights.

Theorem 2.1 ([18]). *Let C be an $[n, k, d]$ linear code. Then:*

1. $1 \leq d_1(C) < d_2(C) < \cdots < d_k(C) \leq n$;
2. (Generalized Singleton Bound) $d_h(C) \leq n - k + h$.

The following classical results from linear algebra and commutative algebra can be found in standard references (see, for instance, [20]).

Corollary 2.1 ([20]). *Let V be a finite–dimensional vector space of dimension n. Any set of n linearly independent vectors in V forms a basis of V.*

Theorem 2.2 ([15]). *Let $A \in \mathrm{Mat}_{n\times n}(\mathbb{K})$. Then A is invertible if and only if $\det(A) \neq 0$.*

Some fundamental results on polynomial rings are presented in [5]. A degree–compatible monomial order is introduced in [11].

Definition 2.2 ([11]). *Let X be a collection of n vector variables $X_1, \ldots, X_n$, where each X_i decomposes into $q-1$ components $x_{i,1}, \ldots, x_{i,q-1}$ for $i \in [n]$. A monomial in X is an expression of the form*

$$X^u = X_1^{u_1} \cdots X_n^{u_n} = \prod_{i=1}^{n} \prod_{j=1}^{q-1} x_{i,j}^{u_{i,j}},$$

where $u \in \mathbb{Z}_{\geq 0}^{n(q-1)}$.

Definition 2.3. *The polynomial ring $\mathbb{K}[X]$ is defined as the set of all polynomials in the variables X with coefficients in the field $\mathbb{K}$.*

The total degree of a monomial X^u is given by $\deg(X^u) = \sum_{i=1}^{n} \sum_{j=1}^{q-1} u_{i,j}$. In particular, when $u = (0, \ldots, 0)$, the monomial reduces to $X^u = 1$.

Let X be a set. A *partial order* on X is a binary relation $\leq$ that is reflexive, antisymmetric, and transitive. A partial order is said to be a *total order* if, for any $a, b \in X$, either $a \leq b$ or $b \leq a$ hold.

Let $\mathbb{K}$ be a field and let $S = \mathbb{K}[x_1, \ldots, x_n]$. Denote by $\mathrm{Mon}(S)$ the set of all monomials in S.

Definition 2.4. *A* monomial order *on S is a total order $\leq$ on $\mathrm{Mon}(S)$ satisfying:*

1. *$1 \leq u$ for all $u \in \mathrm{Mon}(S)$;*
2. *if $u < v$ and $w \in \mathrm{Mon}(S)$, then $uw < vw$.*

Definition 2.5 ([11]). *A monomial order $\prec$ on $\mathbb{K}[x_1, \ldots, x_n]$ is said to be* degree compatible *if*

$$\deg(X^a) < \deg(X^b) \quad \Longrightarrow \quad X^a \prec X^b$$

for all monomials X^a, X^b.

An example of a degree–compatible monomial order on $\mathbb{F}_2[x_1, \ldots, x_n]$ is the graded lexicographic order. It is defined by $X^a <_{\mathrm{deglex}} X^b$ if and only if:

1. $\sum_{i=1}^{n} a_i < \sum_{i=1}^{n} b_i$; or
2. $\sum_{i=1}^{n} a_i = \sum_{i=1}^{n} b_i$ and $a_j < b_j$ for $j = \min\{i \mid a_i \neq b_i\}$.

 doi:10.20944/preprints202602.1166.v1

The following definitions, adapted from [1], formally describe the structure of an information dispersal algorithm.

Definition 2.6 ([1]). *Let $P = \{P_1, P_2, \ldots, P_n\}$ be a set of n participants, and let t be a positive integer called the* threshold. *An* information dispersal algorithm *(IDA) consists of a pair of algorithms* (IDA, IRA) *defined as follows:* **IDA**$(M, P, t) \longrightarrow (X_1, X_2, \ldots, X_n)$, *where:*

- *M denotes the plaintext message;*
- *P is the set of participants;*
- *t is the reconstruction threshold.*

The algorithm produces a collection of shares $(X_1, X_2, \ldots, X_n)$, where each share X_i is distributed to the participant P_i.

IRA$(\{X_i : P_i \in A\}, A)$*:*

- *the input consists of a subset $A \subseteq P$ and the corresponding shares;*
- *if $|A| \geq t$, the algorithm reconstructs M;*
- *otherwise, it produces an error symbol (e.g., `ERROR`).*

The scheme satisfies the following correctness condition:

- (Decoding) *For every subset $A \subseteq P$ with $|A| \geq t$,*

$$\text{IRA}(\{X_i : P_i \in A\}, A) = M.$$

In classical IDA, correctness is ensured by the existence of invertible submatrices. In contrast, our binary variant relies on GHWs and degree compatible monomial orders.

3. An Algebraic Framework for Code-Based IDA/IRA Schemes

Let n be a positive integer and consider the notation $[n] = \{1, \ldots, n\}$. Let C be an $[n, k]$ binary linear code. We define the set $\mathcal{M}$ of codewords of C that form generating sets such that the cardinality of their support equals $d_i(C)$, as follows:

$$\mathcal{M} = \left\{ m \in C \mid \exists\, m_{i_1}, \ldots, m_{i_j} \in C \mid d_i(C) = \left| supp\left(\langle m, m_{i_1}, \ldots, m_{i_j} \rangle \right) \right| \right\},$$

where $i \in \{1, \ldots, k\}$, $i_j \in \{1, \ldots, k-1\}$, and $i = i_j$ if $i \neq 1$ and $i \neq k$.

Observe that for $i = 1$, we have $\{m \in C \mid d_1(C) = |supp(\langle m \rangle)|\} \subseteq \mathcal{M}$, and for $i = k$, $\{m \in C \mid \exists\, m_1, \ldots, m_{k-1} \in C \mid d_k(C) = |supp(\langle m, m_1, \ldots, m_{k-1} \rangle)|\} \subseteq \mathcal{M}$.

Let $\prec$ be a graded monomial order compatible with the total degree in $\mathbb{F}_2[x_1, \ldots, x_n]$. We define the minimal elements in $\mathcal{M}$ as follows:

1. $m^1 = \min_{\prec}(\mathcal{M})$,
2. $m^i = \min_{\prec}\left\{ m \in \mathcal{M} \;\middle|\; d_i(C) = \left| supp\left(\langle m, m^{i_1}, \ldots, m^{i_j} \rangle \right) \right| \right\}$, where $i \in \{2, \ldots, k\}$, $i_j \in \{1, \ldots, k-1\}$, and $i = i_j$ if $i \neq 1$ and $i \neq k$.

Proposition 3.1. *Let C be an $[n, k]$-binary linear code. Then the set $O_{\prec} = \{m^1, \ldots, m^k\}$ is a basis of C.*

Proof. We prove this by induction on the index $i \in [k]$.

For $i = 1$, the codeword m^1 satisfies $d_1(C) = |supp(\langle m^1 \rangle)|$. Clearly, $m^1 \neq 0$, and since $\alpha m^1 = 0$ implies $\alpha = 0$, the vector m^1 is linearly independent.

For illustrative purposes, consider the case $i = 2$. The codewords m^1 and m^2 satisfy $d_2(C) = |supp(\langle m^1, m^2 \rangle)|$. Moreover, $m^1 \neq m^2$ because $d_1 < d_2$, so there is at least one position j where m^2 has a 1 and m^1 has a 0. Let $m^1 = (b_1, \ldots, b_{j-1}, 0, b_{j+1}, \ldots, b_n)$ and $m^2 = (b'_1, \ldots, b'_{j-1}, 1, b'_{j+1}, \ldots, b'_n)$, where $b_i, b'_i \in \{0, 1\}$. Then,

 doi:10.20944/preprints202602.1166.v1

$$\begin{aligned}\alpha_1 m^1 + \alpha_2 m^2 &= \alpha_1(b_1,\ldots,b_{j-1},0,b_{j+1},\ldots,b_n) + \alpha_2(b'_1,\ldots,b'_{j-1},1,b'_{j+1},\ldots,b'_n)\\ &= (\alpha_1 b_1 + \alpha_2 b'_1,\ldots,\alpha_1 b_{j-1} + \alpha_2 b'_{j-1},\alpha_2,\alpha_1 b_{j+1} + \alpha_2 b'_{j+1},\ldots,\alpha_1 b_n + \alpha_2 b'_n)\\ &= (0,\ldots,0).\end{aligned}$$

From this, $\alpha_2 = 0$. Thus,

$$\alpha_1 m^1 + \alpha_2 m^2 = \alpha_1(b_1,\ldots,b_{j-1},0,b_{j+1},\ldots,b_n) = \alpha_1 m^1,$$

and since $\{m^1\}$ is linearly independent, $\alpha_1 = 0$. Therefore, $\alpha_1 = \alpha_2 = 0$, so $\{m^1, m^2\}$ is linearly independent.

Assume that $\{m^1,\ldots,m^{k-1}\}$ is linearly independent in C. Now consider $\{m^1,\ldots,m^k\}$ and suppose $\sum_{i=1}^{k}\alpha_i m^i = 0$. Note that $m^i \neq m^j$ for $i \neq j$ since $d_k(C) = \left|supp(\langle m^1,\ldots,m^k\rangle)\right|$ and $d_1(C) < d_2(C) < \cdots < d_k(C)$. This implies that m^k has at least one 1 in positions $r_1,\ldots,r_j$ where the others have 0s for some $j \in [t]$.

Fix position r_1. The equation becomes

$\sum_{i=1}^{k}\alpha_i m^i = (\sum_{i=1}^{n}\alpha_i b_i,\ldots,\alpha_{r_1} b_{r_1},\ldots,\sum_{i=1}^{n}\alpha_i b_i) = (0,\ldots,0)$,

where $b_i \in \{0,1\}$. From the r_1-th coordinate, $\alpha_{r_1} b_{r_1} = 0$ and since $b_{r_1} = 1$, $\alpha_{r_1} = 0$. Thus, the system reduces to $\sum_{i=1}^{k-1}\alpha_i m^i = 0$. By the induction hypothesis, $\{m^1,\ldots,m^{k-1}\}$ is linearly independent, so $\alpha_i = 0$ for $i \in [k-1]$. Hence, $\alpha_1 = \cdots = \alpha_k = 0$, and $O_\prec = \{m^1,\ldots,m^k\}$ is linearly independent.

By Corollary 2.1, $O_\prec \subset C$ is a basis of C. □

We restate Proposition 3.1 as the following algorithm:

Algorithm 3.1. *Algorithm for computing a basis of a binary linear code.*

Input: An $[n,k]$-binary linear code C and a degree compatible monomial order in the polynomial ring $\mathbb{F}_2[x_1,\ldots,x_n]$.

Output: A basis of C. Operations: A basis of C is obtained as follows:

1. *Compute the codewords in the subset $\mathcal{M}$ of C defined by*

$$\mathcal{M} = \left\{m \in C \mid \exists\, m_{i_1},\ldots,m_{i_j} \in C \mid d_i(C) = \left|supp\Big(\langle m, m_{i_1},\ldots,m_{i_j}\rangle\Big)\right|\right\},$$

   where $i \in \{1,\ldots,k\}$, $i_j \in \{1,\ldots,k-1\}$, and $i = i_j$ whenever $i \neq 1$ and $i \neq k$.
2. *Order the elements of $\mathcal{M}$ according to the order $\prec$ for obtain the sequence of codewords $\{m^1,\ldots,m^k\}$.*
3. *Output: The ordered set of codewords $O_\prec = \{m^1,\ldots,m^k\}$.*

Algorithm 3.2. *Algorithm for IDA using a basis of a binary linear code.*

Input: An $[n,k]$-binary linear code C satisfying $d_1(C) = 1$, $d_k(C) = n$, and a plaintext message M.

Output: A dispersal vector and an $n \times k$ matrix and the threshold t.

Operations: The dispersal vector and the matrix A of size $n \times k$ are obtained as follows:

1. *Apply Algorithm 3.1 to the $[n,k]$-binary linear code C to obtain a basis $O_\prec = \{m^i\}_{i=1}^{k}$ of C.*
2. *Convert the plaintext M into its binary representation, yielding a vector (or a set of vectors) $F \in \mathbb{F}_2^k$.*
3. *Define $A = [m^1, m^2,\ldots,m^k]$. Multiply F (on the right) by A to obtain the dispersal vector d, i.e., $A \cdot F = d$.*
4. *Define $t = n - k$.*
5. *Output: The dispersal vector d, the matrix A of size $n \times k$ and the threshold t.*

Algorithm 3.3. *Algorithm for the recovery of information from a given vector and an associated dispersion matrix, referred to as the Information Recovery Algorithm (IRA).*

 doi:10.20944/preprints202602.1166.v1

Input: A vector b of length k and a matrix $A \in Mat_{n\times k}(\mathbb{F}_2)$.
Output: The dispersion vector d.
Operations: The dispersion vector d is obtained from b and A as follows.

1. *Consider $A = [m^1, m^2, \ldots, m^k]$ and denote by r_i, $i \in [n]$, the rows of A. If $supp(r_i) = supp(r_j)$, remove the row r_j. Repeat this step until all remaining rows satisfy $r_i \neq r_j$, or until $n-k$ deletions are performed. If A contains no repeated rows, proceed to the next step.*
2. *For each $i \in [n]$, if a row $r_i = 0$ in A, remove that row.*
3. *Suppose steps 1 and 2 result in t and s row deletions respectively, with $p, s < n-k$. Identify rows r_i of A such that $|supp(r_i)| \geq |supp(r_j)|$ for all $j \in [n-(p+s)]$, and remove such rows. Repeat until a total of q additional deletions is performed so that $p+s+q = n-k$.*
4. *After deleting $n-k$ rows, rename the resulting column vectors of A as $C_1, C_2, \ldots, C_k$, and define the matrix $B = [C_1, C_2, \ldots, C_k]$.*
5. *Solve the linear system $Bd = b$ for d, i.e., compute $d = B^{-1}b$.*
6. *Output: The dispersion vector d.*

Proposition 3.2. *Let C be an $[n,k]$-binary linear code such that $d_1(C) = 1$ and $d_k(C) = n$. Then Algorithms 3.2 and 3.3 are valid.*

Proof. Consider a basis of C given by Algorithm 3.1, namely $O_\prec = \{m^i\}_{i=1}^k$. Since $d_1 = 1$, the codeword m^1 has exactly one nonzero coordinate in the i-th position; that is, $m^1 = (0, \ldots, 0, \underbrace{1}_{i}, 0, \ldots, 0)$.

Note that the i-th row in matrix A is not repeated because m^1 has a single 1 in position i. Therefore, this row is not omitted. Thus, the vector C_1 of matrix B preserves the unique 1 of m^1 while removing $n-k$ zero entries, so $C_1 = (0, \ldots, 0, \underbrace{1}_{i}, 0, \ldots, 0)$.

Furthermore, Algorithm 3.3 omits zero rows and repeated rows in matrix A, so the rows of B satisfy $r_i \neq r_j$ if $i \neq j$ and $r_i \neq 0$ for $i, j \in [k]$.

Now, suppose $\sum_{i=1}^k \alpha_i r_i = 0$, which gives the system of equations:

$$\begin{cases} \alpha_1 b_{11} + \alpha_2 b_{12} + \cdots + \alpha_k b_{1k} = 0 \\ \alpha_1 b_{21} + \alpha_2 b_{22} + \cdots + \alpha_k b_{2k} = 0 \\ \qquad \vdots \qquad\qquad\qquad\qquad \vdots \\ \alpha_1 b_{k1} + \alpha_2 b_{k2} + \cdots + \alpha_k b_{kk} = 0 \end{cases} \qquad (*)$$

This can be rewritten as $\alpha_1 C_1 + \alpha_2 C_2 + \cdots + \alpha_k C_k = 0$. Note that $C_1 = (0, \ldots, \underbrace{1}_{i}, \ldots, 0)$.

Applying Gaussian elimination to the system,

$$\left(\begin{array}{cccc|c} 0 & b_{12} & \cdots & b_{1k} & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & b_{i2} & \cdots & b_{ik} & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & b_{k2} & \cdots & b_{kk} & 0 \end{array}\right) \sim \left(\begin{array}{cccc|c} 1 & b_{i2} & \cdots & b_{ik} & 0 \\ 0 & b_{22} & \cdots & b_{2k} & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & b_{k2} & \cdots & b_{kk} & 0 \end{array}\right)$$

Since $m^i \neq 0$ and $supp(C_i) \subseteq supp(m^i)$, some b_{ij} in C_i are nonzero. Step 2 of Algorithm 3.3 omits zero rows, and step 3 removes rows with larger supports, leaving at least the row in B where $b_{ij} = 1$ in C_i and $b_{ij} = 0$ in C_{i-1}.

For each column C_i, apply row swaps to place a 1 in position b_{ij}, then eliminate below it. Repeating this k times yields:

 doi:10.20944/preprints202602.1166.v1

$$\left(\begin{array}{cccc|c} 1 & b_{i2} & \cdots & b_{ik} & 0 \\ 0 & 1 & \cdots & b_{2j} & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{array}\right)$$

Thus, system $(*)$ is equivalent to

$$\begin{aligned} \alpha_1 + \alpha_2 b_{12} + \cdots + \alpha_k b_{1k} &= 0 \\ \alpha_2 + \cdots + \alpha_k b_{2j} &= 0 \\ &\vdots \\ \alpha_k &= 0 \end{aligned}$$

From the last equation, $\alpha_k = 0$. From the penultimate, $\alpha_{k-1} + \alpha_k b_{k-1,k} = 0$, so $\alpha_{k-1} = 0$. Back substituting yields $\alpha_k = \alpha_{k-1} = \cdots = \alpha_1 = 0$. Thus, the rows r_i of B are linearly independent over $\mathbb{F}_2^k$.

By Corollary 2.1, B is a basis of $\mathbb{F}_2^k$. Since B has full rank k, it is invertible. □

4. Analysis and Results

This section provides a concrete example to substantiate the algebraic framework established in Section 3. The example illustrates how basis selection, governed by GHWs, results in a generator matrix suitable for the Information Dispersal Algorithm (IDA). The primary objective is to confirm the theoretical consistency of our approach and elucidate the operational procedures, rather than to evaluate empirical performance. The GHWs are calculated following the method of Johnsen and Verdure [6]. For reference, the pseudocode for determining the GHWs of an $[n, k]$-binary linear code is included in Appendix 5.

Example 4.1. *Let C be a $[10, 7]$-binary linear code with generator matrix*

$$G = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{pmatrix}.$$

Using the algorithm described in Appendix A1, the generalized Hamming weights of C are computed as

$$d_1(C) = 1, \quad d_2(C) = 2, \quad d_3(C) = 3, \quad d_4(C) = 5, \quad d_5(C) = 7, \quad d_6(C) = 9, \quad d_7(C) = 10.$$

For each $i \in [k]$, we determine some of the elements of the set $\mathcal{M}$ by selecting the keywords that satisfy the defining condition of this set, namely:

 doi:10.20944/preprints202602.1166.v1

$$
\begin{aligned}
&\text{For } i=1, \left\{ (0000010000), (0010000000), (0100000000) \right\} \subseteq \mathcal{M} \\
&\text{For } i=2, \left\{ (0000010000), (0010000000), (0100000000) \right\} \subseteq \mathcal{M} \\
&\text{For } i=3, \left\{ \begin{array}{l} (0000010000), (0010000000), (0100000000), (0001000100), \\ (0010010000), (0000100001), (0110000000), (0100010000), \dots \end{array} \right\} \subseteq \mathcal{M} \\
&\text{For } i=4, \left\{ \begin{array}{l} (0010010000), (0000100001), (0001000100), \\ (0110000000), (0100010000), (0000001011), \dots \end{array} \right\} \subseteq \mathcal{M} \\
&\text{For } i=5, \left\{ (0001000100), (0000001011), (0000110001), \dots \right\} \subseteq \mathcal{M} \\
&\text{For } i=6, \left\{ (0000001011), (0000110001), (0010100001), \dots \right\} \subseteq \mathcal{M} \\
&\text{For } i=7, \left\{ (1000000110), (0000101010), (1110000110), \dots \right\} \subseteq \mathcal{M}.
\end{aligned}
$$

We consider the lexicographic graded order $\prec$, and using SageMath we calculate the codewords $m^i \in C$, which are:

$$
O_{\prec} = \left\{ \begin{array}{l} m^1 = (0000010000), m^2 = (0010000000), m^3 = (0100000000), m^4 = (0000100001) \\ m^5 = (0001000100), m^6 = (0000001011), m^7 = (1000000110) \end{array} \right\}
$$

Furthermore, by the definition of the set $\mathcal{M}$ we see that:

$$
\begin{aligned}
d_1(C) &= |supp(\left\langle m^1 \right\rangle)| = 1 \\
d_2(C) &= |supp(\left\langle m^1, m^2 \right\rangle)| = 3 \\
&\vdots \\
d_7(C) &= |supp(\left\langle m^1, m^2, m^3, m^4, m^5, m^6, m^7 \right\rangle)| = 10
\end{aligned}
$$

By proposition 3.1, we see that the set $O_{\prec}$ forms a basis of the code C.

We then apply the IDA mechanism proposed by Rabin [14] to the set $O_{\prec}$ as follows:

Let M = "Quantum cryptography is secure" is the plain-text message. We convert the message M into ASCII code, and subsequently into its binary representation,

$$
\begin{aligned}
M = &\ 01010001\ 01110101\ 01100001\ 01101110\ 01110100\ 01110101\ 01101101\ 00100000 \\
&\ 01100011\ 01110010\ 01111001\ 01110000\ 01110100\ 01101111\ 01100111\ 01110010 \\
&\ 01100001\ 01110000\ 01101000\ 01111001\ 00100000\ 01101001\ 01110011\ 00100000 \\
&\ 01110011\ 01100101\ 01100011\ 01110101\ 01110010\ 01100101
\end{aligned}
$$

Let $P = \{P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8, P_9, P_{10}\}$ be a set of 10 participants and let the threshold be defined as $t = n - k = 3$. We consider the matrix formed by the codewords m^i,

 doi:10.20944/preprints202602.1166.v1

$$A = \left(m^1, m^2, m^3, m^4, m^5, m^6, m^7\right) = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

The message M is partitioned into blocks of length 7, and each block is multiplied by A. For illustration, only the first three resulting vectors are shown:

$$D = AM = (d_1, d_2, d_3, d_4, d_5, \dots) = \begin{pmatrix} 0\,1\,0\,0\,1 \\ 0\,1\,0\,1\,1 \\ 1\,0\,1\,0\,1 \\ 0\,1\,1\,1\,0 \\ 1\,1\,1\,0\,0 \\ 0\,1\,0\,0\,1\dots \\ 0\,1\,0\,1\,0 \\ 0\,0\,1\,1\,1 \\ 0\,0\,0\,1\,1 \\ 1\,0\,1\,1\,0 \end{pmatrix}_{10\times\lceil |M|/7\rceil}$$

where $\lceil\cdot\rceil$ denotes the ceiling function, and if the length of the final block is less than 7, it is padded with zeros.

Each component of the resulting vectors is distributed among the participants. We assume that participants P_7, P_8, and P_{10} are unavailable. Let P represent the set of participants used for reconstruction:

$P = \{P_1, P_2, P_3, P_4, P_5, P_6, P_9\}$, satisfying $|P| = 7 \geq t$.

Applying the Information Recovery Algorithm (IRA), the components corresponding to the unavailable participants are removed from each dispersal vector, yielding the vectors f_i. To recover the original message, we construct a submatrix B of the dispersal matrix A by deleting the rows corresponding to the unavailable participants.

By Proposition 3.2, the resulting matrix $B \in \mathbb{F}_2^{7\times 7}$ is invertible. Its inverse, computed using `SageMath`, is explicitly given by

$$B^{-1} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Consequently, the dispersal vectors d_i are recovered by solving the linear system

$BM = f_i$, then, $M = B^{-1} f_i$, so,

 doi:10.20944/preprints202602.1166.v1

$$D = \begin{pmatrix} 0&0&0&0&0&1&0\\ 0&0&1&0&0&0&0\\ 0&1&0&0&0&0&0\\ 0&0&0&0&1&0&0\\ 0&0&0&1&0&0&0\\ 0&0&0&0&0&0&1\\ 1&0&0&0&0&0&0 \end{pmatrix} \begin{pmatrix} 0&1&0&0&1\\ 0&1&0&1&1\\ 1&0&1&0&1\\ 0&1&1&1&0\\ 1&1&1&0&0\\ 0&1&1&1&0\ldots\\ 1&1&1&0&0\\ 0&1&0&0&1\\ 0&1&0&0&1 \end{pmatrix} = \begin{pmatrix} 0\,1\,0\,0\,1\\ 0\,1\,0\,1\,1\\ 1\,0\,1\,0\,1\\ 0\,1\,1\,1\,0\\ 1\,1\,1\,0\,0\\ 0\,1\,0\,0\,1\ldots\\ 0\,1\,0\,1\,0\\ 0\,0\,1\,1\,1\\ 0\,0\,0\,1\,1\\ 1\,0\,1\,1\,0 \end{pmatrix}$$

where i indexes the set of message blocks. This result shows that the original message can be retrieved even if up to $t = n - k$ components are lost in each dispersal vector.

5. Conclusions

The contributions of this work are fundamentally theoretical. We have demonstrated that Generalized Hamming Weights (GHWs) serve as effective algebraic tools for constructing bases of binary linear codes suitable for information dispersal. These results establish sufficient algebraic conditions to guaranty the existence of generator matrices with invertible submatrices over $\mathbb{F}_2$, thereby enabling the application of the Information Dispersal Algorithm (IDA) and the Information Recovery Algorithm (IRA) within a strictly binary framework

This study suggests several avenues for future research, particularly regarding performance and generalized applicability.

Question 5.1. *How does the efficiency of the proposed binary IDA benchmark against classical implementations over finite fields like $GF(2^8)$ and $GF(2^{16})$, as discussed in [17]?*

Question 5.2. *In which parameter regimes (n, k) does the binary IDA/IRA framework offer superior performance or distinct trade-offs compared to traditional approaches over larger fields?*

Furthermore, relaxing the Generalized Hamming Weight conditions presents a compelling direction. It is natural to inquire whether the strict constraints used here can be loosened to accommodate a wider class of binary linear codes.

Question 5.3. *What are the admissible ranges for $d_1(C)$ and $d_k(C)$ (with $d_1(C) > 1$ and $d_k(C) \leq n$) that still permit valid IDA-based dispersal and recovery?*

Addressing these questions would deepen the understanding of the interplay between GHWs, algebraic structures, and information dispersal, potentially paving the way for new families of binary codes suitable for distributed storage and secure data dissemination

Author Contributions: Conceptualization, O.C.-M. and L.P.-L.; methodology, R.M.-J. and O.C.-M.; software, O.C.-M. and L.P.-L; validation, O.C.-M., R.M.-J., R. V.-M. and L.P.-L.; formal analysis, O.C.-M. and L.P.-L.; investigation, O.C.-M. and L.P.-L.; resources, all authors; writing—original draft preparation, O.C.-M., R.V.-M. and L.P.-L.; writing—review and editing, O.C.-M., R.M.-J., R.V.-M., and L.P.-L.; visualization, O.C.-M., R.M.-J, R.V.-M., and L.P.-L.; supervision, L.P.-L.; project administration, L.P.-L.; funding acquisition, R.V.-M., and L.P.-L. All authors have read and agreed to the published version of the manuscript.

Funding: Instituto Politécnico Nacional, [Grant numbers SIP-20250150 (R. Vázquez-Medina)

Acknowledgments: The authors acknowledge the Área de Redes y Telecomunicaciones of the Universidad Autónoma Metropolitana and the Secretaría de Ciencia, Humanidades, Tecnología e Innovación, Mexico, for their support under Grant No. CVU-1082416 (O. Casimiro-Muñoz).

 doi:10.20944/preprints202602.1166.v1

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A Algorithm for Computing the Generalized Hamming Weights of an $[n,k]$-Binary Linear Code

In this appendix, we present an algorithm for computing the generalized Hamming weights (GHWs) of an $[n,k]$–binary linear code. The procedure is based on the algebraic-combinatorial approach introduced by Johnsen and Verdure [6], which relates the GHWs of a linear code to homological invariants of a monomial ideal associated with a matroid. The algorithm is implemented using `SageMath` and exploits the correspondence between the parity-check matrix of the code, the associated binary matroid, and the graded Betti numbers arising from a minimal free resolution of the corresponding circuit ideal.

Algorithm A1 Computation of the GHWs of a binary linear code

Require: Integers n, k
Ensure: Generalized Hamming weights of an $[n,k]$ binary linear code
Input the parameters n and k
Construct a binary linear code $C \subseteq \mathbb{F}_2^n$ with parameters $[n,k]$
Compute a generator matrix G and a parity-check matrix H of C
Construct the binary matroid M represented by the matrix H
Determine the set of circuits $\mathcal{C}$ of the matroid M
Define the polynomial ring $R = \mathbb{F}_2[x_1, \ldots, x_n]$
Construct the circuit ideal $I_{\mathcal{C}} \subseteq R$ generated by the monomials corresponding to the circuits of M
Compute a minimal graded free resolution of $I_{\mathcal{C}}$
Extract the graded Betti numbers and store them in a matrix MC
Initialize an empty list GHW
for each homological degree i in MC **do**
  for each internal degree j in MC **do**
    if $MC_{i,j} \neq 0$ **then**
      Append $i + j$ to GHW
      break
    end if
  end for
end for
return GHW

References

1. Auleriano, I.; Chen, A.; D'olivera, R. Optimal Computational Secret Sharing. *IEEE International Symposium on Information Theory (ISIT)* **2025**, 1-10.
2. Beimel, A. *Secret-Sharing Schemes for General Access Structures: An Introduction*, 1st ed.; University of the Negev, Israel, 2025; 10-97.
3. Dertil, R.; Eren, S. Secret Sharing Schemes Based on Linear Codes over F_2RS. *Asian J. Math Appl.* **2025**, *9*.
4. Deyanira, M. Evaluación de rendimiento del Algoritmo de Dispersión de Información sobre los campos finitos $GF(2^8)$ y $GF(2^{16})$. Master of Science in Information Technology, Autonomous Metropolitan University, CDMX, 2019.
5. García-Marco, I.; Márquez-Corbella, I.; Martínez-Moro, E.; Pitones, Y. Free Resolutions and Generalized Hamming Weights of Binary Linear Codes. *Mathematics*, *2022*, 10, 2079.
6. Johnsen, T.; Verdure, H. Hamming weights and Betti numbers of Stanley-Reisner rings associated to matroids. *Appl. Algebra Eng. Commun. Comput.* **2013**, *24*, 73-93.
7. Johnsen, T.; Verdure, H. Stanley-Reisner resolution of constant weight linear codes. *Designs, Codes and Cryptography* **2012**, *72*.
8. Ling, M.; Xing, C. *Coding Theory: A First Course*, 1st ed.; Cambridge University Press, USA, 2004; 39-66.
9. Li, Mingqiang. On the Confidentiality of Information Dispersal Algorithms and Their Erasure Codes. *ArXiv* **2012**, *abs/1206.4123*.
10. Lin, S. J.; Chung, W. H. An Efficient (n,k) Information Dispersal Algorithm Based on Fermat Number Transforms. *IEEE* **2013**, *8*, 1371-1383.

 doi:10.20944/preprints202602.1166.v1

11. Márquez-Corbella, I.; Martínez-Moro, E.; Suárez-Canedo, E. On the ideal associated to a linear code.*Adv. Math. Commun.* **2016**, *10*, 229–254.
12. Pless, V.; Huffman, W. C. *Fundamentals of Error-Correcting Codes*, 1st ed.; New York, USA, 2003; 1-52.
13. Qian, Q.; Yu, ZT.; Zhang, R.; Hung, CH.A multi-layer information dispersal based encryption algorithm andits application for access control. *Sustainable Computing: Informatics and Systems* **2018**, *20*, 76-87.
14. Rabin, M. O. Efficient dispersal of information for security, load balancing, and fault tolerance. *J. ACM* **1989**, *36*, 335-348.
15. Ron, M. *Introduction to Coding Theory*, 2nd ed.; Cambridge University Press, UK, 2006; pp. 26-47.
16. Van Lint, J.; Van der Geer, G. *Introduction to Coding Theory and Algebraic Geometry*, 1st ed.; Boston, Berlin, USA, Germany, 1998; pp. 11-28.
17. Velázquez, B.; Marcelín, R.; Estudio sobre el desempeño del Algoritmo de Dispersión de Información. *CIAI 2018* **2018**.
18. Wei, V. K. Generalized Hamming weights for linear codes. *IEEE Trans. Inf. Theory* **1991**, *37*, 1412-1418.
19. Wijayanto, A.; Harjito, B. Reduce Rounding Off Errors in Information Dispersal Algorithm. *IC3INA* **2019**, 36-40.
20. Zaldivar, F. *Introducción al álgebra lineal*, 1st ed.; Universidad Nacional Autónoma de México, México, 2019; 1-48.