

Single-deletion recovery in linear codes: a dual-filter criterion and algebraic decoding algorithm

Ivan Horbatiy, and Oleksandr Usatyi

Abstract—Deletion errors pose fundamentally different challenges from substitution errors: when a symbol is removed, all subsequent positions shift, destroying synchronization and making classical syndrome decoding inapplicable. This paper proposes an algebraic algorithm that recovers a single deleted symbol using only the parity-check matrix of any existing linear code over $\text{GF}(q)$, computing the missing symbol by field inversion rather than exhaustive search. A necessary and sufficient condition for unambiguous recovery is proven: two codewords conflict if and only if both a weighted-sum filter and a parity filter fail simultaneously. An offline H-matrix screening algorithm identifies all conflict-free subsets for any parity-check matrix without encoder modification.

Keywords—Decoding; Error correction codes; Galois fields; Parity check codes

I. INTRODUCTION

THE deletion channel – where symbols are removed from a transmitted sequence without substitution – poses fundamentally different challenges from the classical substitution channel. When a symbol is deleted, all subsequent positions shift, destroying the synchronization that standard syndrome decoding relies upon. This distinguishes deletion errors from the well-understood case of bit flips, where the parity-check matrix (H) directly identifies the error location through the syndrome equation.

The problem of single-deletion correction was first solved by Levenshtein [1] and formalized through the Varshamov-Tenengolts (VT) codes [2], [3]. These codes achieve near-optimal redundancy of $\log n + 1$ bits and admit efficient linear-time decoding. However, VT codes are not linear for lengths five and above [4]. No framework exists for determining which subsets of an arbitrary linear code – defined by a given H – are capable of unambiguous single-deletion recovery.

This gap is practically relevant in several application domains. In IoT sensor networks, short linear codes with known parity-check matrices are deployed over lossy channels where symbol deletions occur due to packet fragmentation. In DNA-based data storage, quaternary sequences over $\text{GF}(4)$ suffer from frequent insertion and deletion errors. In barcode and RFID standards, short block codes must tolerate single-symbol deletions without re-transmission. In all these cases, a method

that exploits the existing H for deletion recovery – without modifying the encoder – would be directly applicable.

Recent work on non-binary deletion codes [6], [7] has extended VT constructions to non-binary alphabets, motivated by DNA storage applications where quaternary sequences suffer from frequent insertions and deletions [8], [9]. Codes correcting multiple deletions over q -ary alphabets have also been studied [10], [11], [12], and burst-deletion correction for quaternary codes has been addressed in [13]. However, these constructions require dedicated code design rather than analysis of existing codes.

This paper makes four contributions. First, we propose the Symbolic Syndrome Recovery (SSR) algorithm, which computes the deleted symbol algebraically from the parity equations over a Galois field $\text{GF}(q)$ for any prime-power alphabet size q , reducing complexity by a factor of q compared to brute-force enumeration. Second, we prove a dual-filter criterion establishing the first exact positional characterization of single-deletion conflicts in linear codes. Third, we identify conflict-free codeword subsets for linear codes over arbitrary alphabets and demonstrate the criterion on a $\text{GF}(4)$ example, showing that all classes satisfy the criterion for binary codes of any length. Fourth, we propose the H-matrix screening algorithm (HMS) that pre-identifies all conflict-free VT classes for any given parity-check matrix, enabling deletion recoverability analysis of existing codes without encoder modification. The framework is limited to single-deletion recovery; extension to multiple deletions remains an open problem. The HMS algorithm requires explicit enumeration of all q^k codewords and is therefore applicable to codes with $k \lesssim 30$ information symbols for $\text{GF}(2)$ or $k \lesssim 14$ for $\text{GF}(4)$; the SSR decoding algorithm carries no such restriction and operates in $O(n^2 \cdot r)$ per received word regardless of code size.

II. PRELIMINARIES

A. Linear Codes and the Parity-Check Matrix

Let $\text{GF}(q)$ denote the Galois field of order q , where q is a prime power [14]. An (n, k) linear code C over $\text{GF}(q)$ is a k -dimensional subspace of $\text{GF}(q)^n$. A systematic linear code is defined by its parity-check matrix H of dimensions $r \times n$, where $r = n - k$, such that

$$C = \{x \in \text{GF}(q)^n : H \cdot x = 0\}, \quad (1)$$

This work was supported by Ministry of Education and Science of Ukraine under project No 0124U00082

Authors are with Lviv Polytechnic National University (e-mail: giv@polynet.lviv.ua; oleksandr.a.usatyi@lpnu.ua).



© The Author(s). This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY 4.0, <https://creativecommons.org/licenses/by/4.0/>), which permits use, distribution, and reproduction in any medium, provided that the Article is properly cited.

where all arithmetic is performed over $\text{GF}(q)$. In systematic form, $H = [H_{\text{info}} | I_r]$, where H_{info} is an $r \times k$ matrix over $\text{GF}(q)$ and I_r is the $r \times r$ identity matrix [15], [16]. A codeword $x = (x_1, x_2, \dots, x_n)$ satisfies (1) if and only if its syndrome $s = H \cdot x$ is the zero vector.

B. The Deletion Channel

The single-deletion channel removes exactly one symbol from a transmitted codeword x of length n , producing a received sequence y of length $n - 1$, without indicating which position was deleted [1], [5]. Formally, the deletion of position k from x yields

$$D_k(x) = (x_1, \dots, x_{k-1}, x_{k+1}, \dots, x_n), \quad (2)$$

for $k \in \{1, \dots, n\}$. Unlike substitution errors, a single deletion shifts all subsequent symbol positions, destroying the alignment between received symbols and their original indices. As a consequence, the standard syndrome equation $H \cdot y = 0$ cannot be applied directly to y , since y has length $n - 1$ and the dimensions of H are incompatible [5], [6].

C. Deletion Conflicts and Unique Recovery

Definition 1 (Deletion conflict): Two distinct codewords $x, x' \in C$ are said to produce a deletion conflict at positions (k, k') if $D_k(x) = D_{k'}(x')$. When this occurs, a decoder observing $y = D_k(x)$ cannot determine whether x was transmitted with a deletion at position k , or x' was transmitted with a deletion at position k' .

Definition 2 (Conflict-free set): A subset $S \subseteq C$ is conflict-free if no deletion conflict exists among any two distinct codewords in S for any pair of deletion positions.

A conflict-free set guarantees that single-deletion recovery from any codeword in S is unambiguous, regardless of the deletion position.

D. The Varshamov-Tenengolts Function and VT Classes

Following Levenshtein [1] and Varshamov and Tenengolts [2], define the weighted-sum function for a sequence $x \in \text{GF}(q)^n$ as

$$\text{VT}(x) = \sum_{j=1}^n j \cdot x_j \bmod M, \quad (3)$$

where $M = n(q - 1) + 1$ is an integer modulus and the multiplication $j \cdot x_j$ is performed over the integers. The VT class of index a is defined as

$$\text{VT}_q(a, M) = \{x \in \text{GF}(q)^n : \text{VT}(x) = a\}, \quad (4)$$

for $a \in \{0, 1, \dots, M - 1\}$. The original Varshamov-Tenengolts codes are the binary intersection $\text{VT}_2(a, n + 1) \cap \{0, 1\}^n$, which correct a single deletion with near-optimal redundancy of $\log n + 1$ bits [1], [3], [4]. These codes, however, are not linear for $n \geq 5$ [4], and no general framework exists for analyzing which subsets of an arbitrary linear code defined by H are capable of unambiguous single-deletion recovery.

E. Problem Statement

Given an (n, k) linear code C over $\text{GF}(q)$ defined by a parity-check matrix H , with $M = n(q - 1) + 1$ prime, determine: (i) an algebraic algorithm that computes the deleted symbol directly from H without exhaustive search over the alphabet; and (ii) a necessary and sufficient condition on H and the VT class index a such that $\text{VT}_q(a, M) \cap C$ is conflict-free. This work considers exclusively the single-deletion channel; recovery from two or more simultaneous deletions is outside the

scope of the present analysis. The HMS algorithm (ii) has complexity $O(q^k \cdot n^2 \cdot r)$ and requires full codebook enumeration, making it feasible for $k \leq 30$ ($\text{GF}(2)$) or $k \leq 14$ ($\text{GF}(4)$); the SSR algorithm (i) has complexity $O(n^2 \cdot r)$ and imposes no restriction on code size.

III. SYMBOLIC SYNDROME RECOVERY ALGORITHM

A. Motivation

Given a received sequence y of length $n - 1$, a decoder must identify both the deleted position k and the deleted symbol value. A brute-force approach enumerates all $n \cdot q$ candidate pairs (k, v) , inserts v at position k to form a candidate codeword $\hat{x}$ of length n , and verifies $H \cdot \hat{x} = 0$. This requires $O(n \cdot q \cdot r)$ field operations, where $r = n - k$ is the number of parity-check equations. For large alphabets – such as $\text{GF}(256)$ used in Reed-Solomon codes – this factor of q represents a significant computational burden [15].

B. Algorithm Description

The key observation is that for a fixed candidate deletion position k , the value of the deleted symbol is not unknown in a free sense: it is uniquely determined by the parity-check equations. Let $y = (y_1, \dots, y_{n-1})$ be the received sequence. For each candidate deletion position $k \in \{1, \dots, n\}$, construct the symbolic candidate codeword

$$\hat{x}(k) = (y_1, \dots, y_{k-1}, v_k, y_k, \dots, y_{n-1}), \quad (5)$$

where v_k denotes an unknown symbol at position k .

Substituting $\hat{x}(k)$ into the i -th parity-check equation $H_i \cdot \hat{x}(k) = 0$ gives

$$H[i, k] \cdot v_k = -\sum_{j \neq k} H[i, j] \cdot \hat{x}_j(k), \quad (6)$$

where all arithmetic is over $\text{GF}(q)$. Define the partial syndrome residual for row i and candidate position k as

$$P_i(k) = \sum_{j \neq k} H[i, j] \cdot \hat{x}_j(k) \in \text{GF}(q). \quad (7)$$

If there exists a row i such that $H[i, k] \neq 0$, the deleted symbol value is computed directly by field inversion:

$$v_k = H[i^*, k]^{-1} \cdot P_{i^*}(k) \quad (8)$$

The resulting candidate codeword $\hat{x}(k)$ is then verified against all remaining parity-check rows. If all r equations are satisfied, $\hat{x}(k)$ is accepted as a valid candidate. Algorithm 1 formalizes this procedure.

Remark 1: The negative sign in (6) is immaterial when $\text{char}(\text{GF}(q)) = 2$, since $-x = x$ for all $x \in \text{GF}(q)$. All fields considered in this paper satisfy this condition ($q \in \{2, 4, 8, 256\}$). For fields of odd characteristic, equation (8) should read $v_k = -H[i^*, k]^{-1} \cdot P_{i^*}(k)$.

Algorithm 1

Symbolic Syndrome Recovery (SSR)

Input: received sequence y in $\text{GF}(q)^{(n-1)}$,

parity-check matrix H in $\text{GF}(q^{*n})$

Output: set of candidate codewords $S \subseteq \text{GF}(q)^n$

```

S = {}
for k = 1 to n do
  Compute P_i(k) for all i = 1, ..., r using (7)
  Find any row i* such that H[i*, k] <> 0
  if no such row exists then skip
  Compute v = H[i*, k]^{-1} * P_i*(k) over GF(q)
  Construct x_hat(k) by inserting v at position k
in y
  if H * x_hat(k) = 0 then S = S + {x_hat(k)}
end for
return S

```

C. Complexity Analysis

For each of the n candidate positions, computing $P_i(k)$ requires $O(r \cdot n)$ field multiplications. The field inversion in step 6 costs $O(1)$ for fixed q (via a precomputed table over $GF(q)$). The verification in step 8 costs $O(r \cdot n)$. The total complexity of Algorithm 1 is therefore

$$T_{SSR} = O(n^2 \cdot r), \quad (9)$$

where r is the number of parity-check equations. In contrast, brute-force enumeration evaluates all q possible values of the deleted symbol at each of the n positions, giving

$$T_{BF} = O(n^2 \cdot r \cdot q). \quad (10)$$

The fundamental difference between the two approaches is the number of symbol values tested per candidate position. Brute-force tests all q values of the alphabet at each of the n positions – most of which are guaranteed to be wrong – yielding $n \cdot q$ candidate codewords to verify. SSR instead computes exactly one candidate value per position through direct field inversion: if $H[i, k] \neq 0$, then the parity equation uniquely determines $v = H[i, k]^{-1} \cdot P_i(k)$. This eliminates q^{-1} unnecessary verifications at each position. The SSR algorithm thus achieves a factor-of- q reduction in complexity over brute force. For binary codes ($q=2$) this yields a $2\times$ improvement; for $GF(4)$ a $4\times$ improvement; and for $GF(256)$ – used in Reed-Solomon codes – the improvement reaches $256\times$. The advantage grows with alphabet size, making SSR particularly beneficial for non-binary codes over large fields.

D. Correctness

Proposition 1: Algorithm 1 never misses the correct codeword. That is, if x was transmitted and symbol x_k was deleted, then $x \in S$.

Proof: Since $x \in C$, we have $H \cdot x = 0$. Inserting x_k at position k in y reconstructs x exactly. The residual $P_{i^*}(k)$ computed from y equals $\sum_{j \neq k} H[i^*, j] \cdot x_j$, so $H[i^*, k]^{-1} \cdot P_{i^*}(k) = x_k$. Since H is in systematic form $H = [H_{\text{info}} | I_r]$, at least one row i^* satisfies $H[i^*, k] \neq 0$ for any column k , guaranteeing the inversion is always defined. The verification step confirms $H \cdot x = 0$, so x is added to S .

The set S may contain more than one candidate when two distinct codewords produce the same sequence after deleting different positions. This is precisely the deletion conflict defined in Section II. Algorithm 1 is complete (0% failed recoveries) but may be ambiguous when $|S| > 1$. The condition under which $|S| = 1$ is guaranteed for all inputs from a given VT class is the subject of Section IV.

IV. NECESSARY AND SUFFICIENT CONDITION FOR UNIQUE RECOVERY

A. The Shift Operation

To analyze when two distinct codewords produce a deletion conflict, we introduce the shift operation. Let $x \in C$ and let $1 \leq k_1 < k_2 \leq n$. The shift of x at positions (k_1, k_2) with inserted symbol value $v \in GF(q)$ is defined as

$$\text{shift}(x, k_1, k_2, v) = (x_1, \dots, x_{k_1-1}, x_{k_1+1}, \dots, x_{k_2}, v, x_{k_2+1}, \dots, x_n) \quad (11)$$

The sequence $\text{shift}(x, k_1, k_2, v)$ is obtained by deleting position k_1 from x and inserting symbol v at position k_2 . A deletion conflict between x and a second codeword $x' \in C$ exists at positions k_1, k_2 if and only if $\text{shift}(x, k_1, k_2, v) = x'$ for some $v \in GF(q)$, i.e., if and only if $\text{shift}(x, k_1, k_2, v) \in C$ for some $v \in \{0, \dots, q-1\}$.

B. Three Analytical Lemmas

Lemma 1 (Parity condition for shift closure): Let $x \in C$, $1 \leq k_1 < k_2 \leq n$, and $v \in GF(q)$. Define the residual for row i of H as

$$R_i(x, k_1, k_2) = \sum_{j=k_1}^{k_2} H[i, j] \cdot x_j + \sum_{j=k_1}^{k_2-1} H[i, j] \cdot x_{j+1} \quad (12)$$

where $\oplus$ denotes addition in $GF(q)$. Then $\text{shift}(x, k_1, k_2, v) \in C$ if and only if

$$R_i(x, k_1, k_2) + H[i, k_2] \cdot v = 0 \quad \forall i = 1, \dots, r \quad (13)$$

Proof: Since $x \in C$, we have $H \cdot x = 0$. Let $x' = \text{shift}(x, k_1, k_2, v)$. Computing $H \cdot x'$ row by row and subtracting $H \cdot x = 0$ yields equation (13) directly. Explicitly, for each row i , the j -th term of $H \cdot x'$ is $H[i, j] \cdot x'_j$. For $j < k_1$ or $j > k_2$, $x'_j = x_j$, so these terms cancel with $H \cdot x = 0$. For $k_1 \leq j \leq k_2-1$, $x'_j = x_{j+1}$, contributing $H[i, j] \cdot x_{j+1}$. The inserted symbol v appears at position k_2 with coefficient $H[i, k_2]$. Collecting all window terms and subtracting $H \cdot x = 0$ gives $R_i(x, k_1, k_2) + H[i, k_2] \cdot v = 0$, which is equation (13).

In equation (12), the first sum captures the original symbol contributions at positions k_1 through k_2 in x , while the second sum captures the shifted contributions, where each x_{j+1} replaces x_j due to the left-shift caused by the deletion at k_1 . The two sums together quantify the net change in the syndrome when the block of symbols between k_1 and k_2 is shifted by one position.

Numerical illustration: Lemma 1 was exhaustively verified on all 15,360 combinations of codeword, position pair, and symbol value for the example $GF(4)$ code of Section V ($n=6$, $q=4$, $M=19$), with 100% agreement.

Lemma 2 (VT value of a shifted codeword): Let $x \in C$ with $VT(x) = a$, and let $\Delta(x, k_1, k_2)$ denote the weighted-sum difference

$$\Delta(x, k_1, k_2) = \sum_{j=k_1}^{k_2} j \cdot x_j - \sum_{j=k_1}^{k_2-1} j \cdot x_{j+1}, \quad (14)$$

computed over the integers. Then

$$VT(\text{shift}(x, k_1, k_2, v)) = a - \Delta(x, k_1, k_2) + k_2 \cdot v \mod M \quad (15)$$

Proof: Expand $VT(\text{shift}(x, k_1, k_2, v))$ by splitting the sum into prefix, window, and suffix contributions. The prefix and suffix terms equal the corresponding terms of $VT(x)$. The window contribution changes from $\sum_{j=k_1}^{k_2} j \cdot x_j$ to $\sum_{j=k_1}^{k_2-1} j \cdot x_{j+1} + k_2 \cdot v$, yielding (15).

Numerical illustration: Lemma 2 was verified on 3,000 randomly selected combinations from the example $GF(4)$ code of Section V, with 100% agreement.

Lemma 3 (Unique VT candidate when M is prime): Let $M = n(q-1) + 1$ be prime. A conflict between $x \in C$ and a codeword in the same VT class requires $VT(\text{shift}(x, k_1, k_2, v)) = a$, which by (15) reduces to

$$k_2 \cdot v \equiv \Delta(x, k_1, k_2) \mod M. \quad (16)$$

Since M is prime and $k_2 \in \{1, \dots, n\} \subset \{1, \dots, M-1\}$, the modular inverse $k_2^{-1} \mod M$ exists for all valid k_2 . Equation (16) therefore has the unique integer solution

$$v^* = k_2^{-1} \cdot \Delta(x, k_1, k_2) \mod M. \quad (17)$$

A conflict is possible only if $v^* \in \{0, 1, \dots, q-1\}$, i.e., only if v^* falls within the alphabet.

C. Main Theorem

Theorem 1 (Dual-filter criterion): Let C be a linear code over $GF(q)$ of length n with parity-check matrix H , and let $M = n(q-1) + 1$ be prime. The set $VTq(a, M) \cap C$ is conflict-free if and only if for every codeword $x \in VTq(a, M) \cap C$ and every pair of positions $1 \leq k_1 < k_2 \leq n$, at least one of the following conditions holds:

(Case A – VT filter): $v^* = k_2^{-1} \cdot \Delta(x, k_1, k_2) \bmod M$ satisfies $v^* \notin \{0, 1, \dots, q-1\}$;

(Case B – Parity filter): $v^* \in \{0, 1, \dots, q-1\}$ but $H \cdot \text{shift}(x, k_1, k_2, v^*) \neq 0$.

Proof: A conflict at (x, k_1, k_2) exists if and only if there exists $v \in \{0, \dots, q-1\}$ satisfying both Lemma 1 and the VT-preservation condition $VT(\text{shift}) = a$. By Lemma 2, the VT condition reduces to equation (16). By Lemma 3, since M is prime, equation (16) has the unique solution v^* given by (17). If $v^* \notin \{0, \dots, q-1\}$, no alphabet symbol can satisfy the VT condition, so no conflict is possible – this is Case A. If $v^* \in \{0, \dots, q-1\}$, a conflict exists if and only if v^* also satisfies Lemma 1, i.e., $\text{shift}(x, k_1, k_2, v^*) \in C$ – this is Case B. The set $VTq(a, M) \cap C$ is conflict-free if and only if Case A or Case B eliminates every potential conflict for every (x, k_1, k_2) .

Remark 2: The primality of M is essential. If M is composite, equation (16) may have zero, one, or multiple solutions depending on $\gcd(k_2, M)$, and the two-case structure of Theorem 1 does not apply in general.

Remark 3: Case A depends only on the codeword values and position indices, not on H . Case B depends on the structure of H . The theorem thus separates the contribution of the VT weighting from the contribution of the linear code structure.

Numerical illustration: Theorem 1 was verified exhaustively on all 3,840 (codeword, position pair) combinations for the example GF(4) code of Section V ($n=6, q=4, M=19$), with 100% agreement between the theorem prediction and direct conflict enumeration.

V. RESULTS

A. Code Parameters and Experimental Setup

The proposed algorithm and theorem are evaluated on a systematic linear code C over $GF(4)$ of length $n=6$ and dimension $k=4$, defined by the parity-check matrix

$$H = [H_{info} | I_2], \quad (18)$$

where

$$H_{info} = [[1, 0, 1, 1], [0, 1, 1, 2]] \quad (19)$$

and I_2 is the 2×2 identity matrix, with all entries in $GF(4) = \{0, 1, \alpha, \alpha+1\}$ represented as $\{0, 1, 2, 3\}$. The code contains $q^k = 4^4 = 256$ codewords. The VT modulus is $M = n(q-1) + 1 = 6 \cdot 3 + 1 = 19$, which is prime, satisfying the condition of Theorem 1.

B. VT Class Structure

The 256 codewords are distributed across $M = 19$ VT classes. All 19 classes are non-empty. Applying the dual-filter criterion of Theorem 1 exhaustively to all codewords and all position pairs (k_1, k_2) , four classes are identified as conflict-free. The results are summarized in TABLE I.

TABLE I.
VT CLASS ANALYSIS FOR THE GF(4) CODE ($N=6, Q=4, M=19$)

VT CLASS INDEX A	CLASS SIZE	CASE A INSTANCES	CASE B CONFLICTS	CONFLICT-FREE
0	14	158	1	No
1	13	150	3	No
2	10	106	1	No
3	22	255	5	No
4	12	124	2	No
5	6	65	0	Yes
6	16	182	3	No
7	24	289	5	No
8	6	57	0	Yes
9	12	139	6	No
10	14	164	1	No
11	18	211	1	No
12	10	116	1	No
13	9	93	0	Yes
14	16	187	1	No
15	18	209	3	No
16	7	78	0	Yes
17	12	133	2	No
18	17	197	1	No

C. Conflict-Free Classes

The four conflict-free classes are $VTq_4(a, 19) \cap C$ for $a \in \{5, 8, 13, 16\}$, representing 21% of all non-empty VT classes. Unambiguous single-deletion recovery is guaranteed when transmitted codewords are restricted to any single conflict-free class. The largest such class ($a=13$) contains 9 codewords, supporting transmission of $\lfloor \log_2 9 \rfloor = 3$ information bits per codeword. For each codeword in a chosen conflict-free class, every candidate deletion position $k \in \{1, \dots, n\}$ ($n=6$ here) yields a unique candidate from Algorithm 1, i.e., $|S| = 1$ for all inputs. This guarantees unambiguous single-deletion recovery without knowledge of the originally transmitted codeword.

D. Binary Case

For binary linear codes ($q = 2$), the SSR algorithm was evaluated on systematic codes of lengths $n \in \{4, 6, 10\}$ with prime moduli $M \in \{5, 7, 11\}$ respectively. In all cases examined, every non-empty VT class was found to be conflict-free.

This empirical observation – that all VT classes are conflict-free for binary linear codes with prime M – is stated here as a conjecture and is outside the scope of the present analytical framework. No counterexample was found for $n \in \{4, 6, 10\}$, but a formal proof remains open. The result does not follow directly from Theorem 1, as it would require characterising the distribution of $\Delta(x, k_1, k_2) \bmod M$ over all binary codewords of a given linear code.

E. Theorem Verification

Theorem 1 was verified exhaustively against direct conflict enumeration on the complete set of 3,840 (codeword, position pair) combinations for the example GF(4) code ($n=6, q=4, M=19$), with 3,840 out of 3,840 predictions correct (100%). The verification confirms that the dual-filter criterion is both necessary and sufficient: every conflict detected by direct enumeration corresponds to a Case B failure in Theorem 1, and every Case B failure corresponds to an actual conflict.

F. H-Matrix Screening for Deletion Recoverability

Theorem 1 enables a pre-transmission screening procedure: given any parity-check matrix H , one can determine analytically which VT classes support unambiguous single-deletion recovery before any transmission takes place. Algorithm 2 formalizes this screening.

Algorithm 2: H-Matrix Screening (HMS)

Input: parity-check matrix H , code C , modulus $M = n \cdot (q-1) + 1$ (prime)

Output: set of conflict-free VT class indices $A \subseteq \{0, 1, \dots, M-1\}$

```

Compute VT(x) for all x in C
Partition C into classes  $VT_q(a, M) \cap C$  for  $a = 0, \dots, M-1$ 
For each class index a:
    For each codeword x in  $VT_q(a)$  and each pair
    1 ≤ k1 < k2 ≤ n:
        Compute  $v^* = k2^{(-1)} \cdot \Delta(x, k1, k2)$ 
        mod M via (17)
        If  $v^* \in \{0, \dots, q-1\}$  and  $H \cdot \text{shift}(x, k1, k2, v^*) = 0$ :
            mark class a as dirty
            break
A = {a in {0, ..., M-1} : class a was not marked dirty}
return A

```

The screening complexity is $O(|C| \cdot n^2 \cdot r)$, executed once offline per code. Once A is known, an encoder assigns transmitted codewords exclusively to classes in A , and the SSR decoder is guaranteed to return $|S| = 1$ for every received sequence. This pre-screening separates code design from deletion recoverability analysis: any existing linear code with an enumerable codebook can be screened without modifying its encoder. For the GF(4) code defined by (18)–(19), HMS identifies four conflict-free VT classes. The largest class ($a = 13$) contains 9 codewords out of 256 total, supporting transmission of $\lceil \log_2 9 \rceil = 3$ information bits per codeword, compared to 8 bits available in the full code. The choice of H critically determines how many conflict-free codewords exist: the present example achieves 9 out of 256, while a different H over the same parameters may yield a larger or smaller conflict-free subset.

VI. DISCUSSION

A. Interpretation of the Dual-Filter Structure

Theorem 1 identifies two independent mechanisms that prevent deletion conflicts. Case A – the VT filter – eliminates conflicts through a purely arithmetic condition on the codeword symbol values and position indices, independently of the parity-check matrix H . Case B – the parity filter – eliminates conflicts through the algebraic structure of H . This separation is practically significant: it implies that the conflict-free property of a VT class is jointly determined by the choice of VT modulus M and the design of H , and that either factor alone may be sufficient to prevent ambiguity for a given position pair.

B. Comparison with Related Approaches

The proposed SSR algorithm occupies a distinct position relative to existing single-deletion correction methods. Classical Varshamov–Tenengolts (VT) codes [1]–[3] achieve near-optimal redundancy of $\log_2 n + 1$ bits and admit linear-time decoding, but require a dedicated code construction: the encoder

must be redesigned to produce codewords satisfying the VT checksum constraint. VT codes are also non-linear for $n \geq 5$ [4], which precludes their use within existing linear coding frameworks. By contrast, SSR operates on any linear code defined by an arbitrary H without modifying the encoder; the conflict-free subset is identified offline by HMS and used as a transmission constraint.

Marker codes and watermark-based methods [17] address deletion correction in streaming or concatenated settings by embedding synchronisation patterns into the codestream. These approaches target burst deletions and multi-deletion channels, and their decoding relies on the presence of the embedded markers rather than on the algebraic structure of a parity-check matrix. SSR, by contrast, exploits the existing H directly and targets the single-deletion channel for block codes.

In terms of decoding complexity, VT codes decode in $O(n)$ [1], while SSR requires $O(n^2 \cdot r)$. This is a direct consequence of the more general setting: VT codes exploit a global checksum that uniquely identifies the deletion position, whereas SSR must test all n candidate positions without such a checksum. The factor-of- q speedup of SSR over brute-force enumeration is therefore measured relative to the natural baseline for the H -based setting, not relative to purpose-built deletion codes. The key contribution is not complexity superiority over VT codes, but rather the ability to perform deletion recovery within an existing linear code without encoder modification.

C. Role of the Primality Condition

The primality of $M = n(q-1) + 1$ is a necessary condition for Theorem 1 to hold in its two-case form. When M is prime, equation (16) has a unique solution v^* for every position pair (k_1, k_2) and every codeword, which makes the analysis tractable. When M is composite, the equation $k_2 \cdot v \equiv \Delta \pmod{M}$ may have zero or multiple solutions depending on $\gcd(k_2, M)$, and the conflict condition becomes substantially more complex. For the parameters examined in this work – $n = 6$, $q = 4$, $M = 19$ – the primality condition is satisfied. Pairs (n, q) for which $M = n(q-1) + 1$ is prime include $(n=4, q=2, M=5)$, $(n=6, q=2, M=7)$, $(n=6, q=4, M=19)$, and $(n=10, q=2, M=11)$, among others.

D. Application to Existing Communication Standards

The proposed framework applies at two levels depending on code size. For short linear codes where explicit codeword enumeration is feasible – including IoT sensor codes, DNA storage codes over GF(4), and short block codes used in barcode standards – both SSR and HMS are directly applicable. The SSR algorithm recovers the deleted symbol with complexity $O(n^2 \cdot r)$, and HMS identifies all conflict-free VT classes offline in $O(|C| \cdot n^2 \cdot r)$ operations. For codes with k up to approximately 16 information symbols over GF(4), the full pipeline runs in milliseconds on standard hardware.

The HMS pre-screening algorithm requires enumerating all q^k codewords, which limits its applicability to codes with $k \leq 30$ for GF(2) or $k \leq 14$ for GF(4) on standard hardware. For codes within these bounds – including short IoT block codes, DNA storage codes over GF(4), and barcode standards – the full pipeline of HMS followed by SSR decoding runs in milliseconds. For larger codes, HMS in its current form is computationally infeasible; an efficient variant that avoids full codebook enumeration remains an open problem.

E. VT Class Size and Rate Loss Analysis

A fundamental question for practical deployment is how many codewords a conflict-free VT class can contain, and how much rate is sacrificed by restricting transmission to such a class. TABLE II provides a numerical analysis across a range of code parameters (n, q, k) with prime modulus $M = n(q - 1) + 1$. For each configuration, the maximum VT class size is computed over all codewords in C , and the rate loss is expressed in information bits relative to the full code capacity $k \cdot \log_2 q$ bits.

TABLE II
VT CLASS SIZE AND RATE LOSS FOR LINEAR CODES WITH PRIME M

N	Q	K	M	$ C =QK$	MAX CLASS SIZE	BITS (FULL)	BITS (UPPER BOUND)	RATE LOSS, %
4	2	2	5	4	2	2	1	50.0
6	2	4	7	16	4	4	2	50.0
10	2	8	11	256	26	8	4	50.0
12	2	10	13	1,024	82	10	6	40.0
16	2	14	17	16,384	968	14	9	35.7
6	4	4	19	256	24	8	4	50.0
10	4	8	31	65,536	2,381	16	11	31.2
6	8	4	43	4,096	122	12	6	50.0

Max class size: largest VT class among all codewords in C , not restricted to conflict-free classes. Bits (upper bound) = $\lceil \log_2(\text{max class size}) \rceil$ is the theoretical maximum achievable by an optimal H . The actual bits supported by a specific H may be lower; for the paper example ($n=6, q=4$), the largest CF class under H of (18)–(19) contains 9 codewords (3 bits, TABLE I). Rate loss = (bits full – bits upper bound) / bits full.

TABLE II reveals an important trend: rate loss is bounded asymptotically by $\log_2 M = \log_2(n(q - 1) + 1)$ bits, which is independent of k . For large k , this fixed overhead becomes a negligible fraction of the total code capacity $k \cdot \log_2 q$, and the effective code rate approaches the full code rate. This matches Levenshtein's near-optimal redundancy bound of $\log_2 n + 1$ bits for binary single-deletion correction [1].

F. Open Problems

Two analytical questions raised by the results of Section V remain unresolved. First, the empirical finding that all VT classes are conflict-free for binary linear codes with prime M (Section V, Binary Case) lacks an analytical proof. The intuitive argument is that the binary alphabet $\{0,1\}$ causes v^* to fall outside the alphabet in most cases. We conjecture that Case A is always sufficient for $q=2$, but this remains unproven; a formal proof would require characterising the distribution of $\Delta(x, k_1, k_2) \bmod M$ over all binary codewords.

Second, a closed-form characterization of which parity-check matrices H yield conflict-free VT classes for given code parameters remains open. Exhaustive search over $\text{GF}(4)$ with $n=6$ shows that 92 out of 64,260 valid H matrices yield all 19 VT classes conflict-free, but no algebraic condition distinguishing these matrices has been identified.

G. Complexity and Practical Considerations

The one-time pre-verification cost of identifying all conflict-free VT classes for a given code is $O(|C| \cdot n^2 \cdot r)$ field

operations, as described in the complexity analysis of Section III. Once the conflict-free classes are identified, the per-codeword decoding cost of SSR is $O(n^2 \cdot r)$, which is polynomial in the code length and independent of the alphabet size. For codes where the transmitted codeword is known to belong to a conflict-free VT class – for instance, through an encoding constraint – SSR provides guaranteed unique recovery with complexity lower than brute-force by a factor of q [15], [17]. This advantage grows with alphabet size and is most significant for non-binary codes: for binary codes ($q=2$) the speedup is $2\times$; for $\text{GF}(4)$ codes used in DNA storage it is $4\times$; and for $\text{GF}(256)$ codes used in Reed-Solomon error correction the speedup reaches $256\times$. For a Reed-Solomon code with $n=255$ over $\text{GF}(256)$, brute-force requires $255 \times 256 \times r = 65,280 \cdot r$ field operations per received word, while SSR requires only $255 \times r$ operations – a reduction that is practically significant for real-time decoding applications.

VII. CONCLUSIONS

This paper presented four novel contributions to the problem of single-symbol deletion recovery in linear codes defined by an arbitrary parity-check matrix H .

First, the Symbolic Syndrome Recovery (SSR) algorithm was proposed – the first algebraic decoding method for single-deletion recovery that operates on any existing linear code over $\text{GF}(q)$ without encoder modification. By treating the deleted symbol as an unknown and solving the parity-check equations algebraically over $\text{GF}(q)$, SSR computes the deleted symbol value directly through field inversion rather than exhaustive enumeration. The complexity of SSR is $O(n^2 \cdot r)$, a factor of q lower than brute-force search, with 0% failed recoveries guaranteed by construction.

Second, for the first time, a necessary and sufficient condition for unambiguous single-deletion recovery in linear codes over $\text{GF}(q)$ was proven through three analytical lemmas. The dual-filter criterion of Theorem 1 establishes that a VT class $VT_q(a, M) \cap C$ is conflict-free if and only if for every codeword and every pair of deletion positions, either the VT filter (Case A) or the parity filter (Case B) eliminates the potential conflict. The proof relies on the primality of $M = n(q - 1) + 1$, which guarantees a unique candidate symbol value per position pair. The theorem was exhaustively verified on 3,840 cases with 100% agreement.

Third, the dual-filter criterion was applied to linear codes over arbitrary alphabets $\text{GF}(q)$, providing the first characterisation of conflict-free subsets for non-binary linear codes. As a concrete numerical example, a systematic $\text{GF}(4)$ linear code of length $n = 6$ was analysed. Four conflict-free VT classes were identified; the largest contains 9 codewords, supporting 3 information bits per codeword when transmission is restricted to a single conflict-free class. For binary linear codes of lengths $n \in \{4, 6, 10\}$ with prime M , all VT classes were empirically found to be conflict-free, motivating an open analytical question about the structural properties of binary codewords modulo M . All results in this paper concern single-deletion recovery only; the case of two or more deletions remains an open problem.

Fourth, the H-matrix screening algorithm (HMS, Algorithm 2) was proposed as the first offline pre-transmission tool that identifies all conflict-free VT classes for any given parity-check

matrix H in $O(|C| \cdot n^2 \cdot r)$ operations. For the GF(4) code defined by (18)–(19), HMS identified four conflict-free VT classes. The largest class contains 9 codewords out of 256, enabling transmission of $\lfloor \log_2 9 \rfloor = 3$ information bits per codeword with guaranteed single-deletion recovery, compared to 8 bits available in the full code. This demonstrates that the choice of H critically determines deletion recoverability, and that HMS provides a practical tool for evaluating this property for any given parity-check matrix. HMS has complexity $O(q^k \cdot n^2 \cdot r)$ and is feasible for $k \lesssim 30$ (GF(2)) or $k \lesssim 14$ (GF(4)); extension to larger codes requires an efficient variant without full codebook enumeration.

The results demonstrate that the parity-check matrix of an existing linear code carries sufficient algebraic structure to support single-deletion recovery for specific subsets of codewords, without any modification to the encoder. This is directly applicable to short linear codes deployed in IoT sensor networks, DNA-based data storage over GF(4), and barcode standards, where single-symbol deletions occur and retransmission is costly or unavailable.

The primary novelty of this work lies in establishing that single-deletion recovery is achievable within any existing linear code – without dedicated construction – through algebraic exploitation of the parity-check matrix H . Future work includes an analytical characterisation of why binary codes empirically exhibit universal conflict-freeness, a closed-form characterization of which VT class indices yield conflict-free classes for a given H over GF(q), an extension of the dual-filter analysis to the composite M case and to multiple deletions, an optimization framework for constructing H matrices that maximize the number of conflict-free VT classes, and an efficient HMS variant that avoids explicit codeword enumeration and extends applicability beyond the current $k \lesssim 30$ (GF(2)) and $k \lesssim 14$ (GF(4)) bounds.

REFERENCES

- [1] V. I. Levenshtein, Binary codes capable of correcting deletions, insertions and reversals, *Soviet Physics Doklady*, vol. 10, no. 8, pp. 707–710, 1966.
- [2] R. R. Varshamov and G. M. Tenengolts, A code for correcting a single asymmetric error, *Automation and Remote Control*, vol. 26, no. 2, pp. 286–290, 1965.
- [3] G. Tenengolts, Nonbinary codes, correcting single deletion or insertion, *IEEE Trans. Inf. Theory*, vol. 30, no. 5, pp. 766–769, 1984. DOI: 10.1109/TIT.1984.1056962
- [4] N. J. A. Sloane, On single-deletion-correcting codes, in *Codes and Designs*, K. T. Arasu and A. Seress, Eds. Berlin: De Gruyter, 2002, pp. 273–292. DOI: 10.1515/9783110198119.273
- [5] H. Mercier, V. K. Bhargava, and V. Tarokh, A survey of error-correcting codes for channels with symbol synchronization errors, *IEEE Commun. Surveys Tuts.*, vol. 12, no. 1, pp. 87–96, 2010. DOI: 10.1109/SURV.2010.020110.00079
- [6] B. Haeupler and A. Shahrasbi, Synchronization strings and codes for insertions and deletions – a survey, *IEEE Trans. Inf. Theory*, vol. 67, no. 6, pp. 3190–3206, 2021. DOI: 10.1109/TIT.2021.3056317
- [7] J. Sima and J. Bruck, On optimal k -deletion correcting codes, *IEEE Trans. Inf. Theory*, vol. 67, no. 6, pp. 3360–3375, 2021. DOI: 10.1109/TIT.2020.3028702
- [8] V. Guruswami and J. Håstad, Explicit two-deletion codes with redundancy matching the existential bound, *IEEE Trans. Inf. Theory*, vol. 67, no. 10, pp. 6384–6394, 2021. DOI: 10.1109/TIT.2021.3069446
- [9] I. Shomorony and R. Heckel, Information-theoretic foundations of DNA data storage, *Foundations and Trends in Communications and Information Theory*, vol. 19, no. 1, pp. 1–106, 2022. DOI: 10.1561/0100000107
- [10] R. Gabrys, V. Guruswami, J. Ribeiro, and K. Wu, Beyond single-deletion correcting codes: substitutions and transpositions, *IEEE Trans. Inf. Theory*, vol. 69, no. 1, pp. 169–186, 2023. DOI: 10.1109/TIT.2022.3201207
- [11] T. T. Nguyen, K. Cai, and P. H. Siegel, A new version of q -ary Varshamov-Tenengolts codes with more efficient encoders: the differential VT codes and the differential shifted VT codes, *IEEE Trans. Inf. Theory*, vol. 70, no. 10, pp. 6989–7004, Oct. 2024. DOI: 10.1109/TIT.2024.3417894
- [12] K. Cai, Y. M. Chee, R. Gabrys, H. M. Kiah, and T. T. Nguyen, Correcting a single indel/edit for DNA-based data storage: linear-time encoders and order-optimality, *IEEE Trans. Inf. Theory*, vol. 67, no. 6, pp. 3438–3451, Jun. 2021. DOI: 10.1109/TIT.2021.3049627
- [13] T.-H. Khuat and S. Kim, A quaternary code correcting a burst of at most two deletion or insertion errors in DNA storage, *Entropy*, vol. 23, no. 12, p. 1592, Nov. 2021. DOI: 10.3390/e23121592
- [14] R. Lidl and H. Niederreiter, *Finite Fields*, 2nd ed. Cambridge: Cambridge University Press, 1997.
- [15] S. Lin and D. J. Costello, *Error Control Coding*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 2004.
- [16] R. M. Roth, *Introduction to Coding Theory*. Cambridge: Cambridge University Press, 2006.
- [17] T. Xue, Y. Wang, F. C. M. Lau, and C. Yi, A concatenated LDPC-marker code for channels with correlated insertion and deletion errors in bit-patterned media recording system, *PLOS ONE*, vol. 17, no. 7, p. e0270247, Jul. 2022. DOI: 10.1371/journal.pone.0270247