X-Codes: Theory and Applications of Unknowable Inputs

Steven S. Lumetta
Department of ECE
University of Illinois
lumetta@uiuc.edu

Subhasish Mitra
Intel Corporation
subhasish.mitra@intel.com

1 Introduction

This paper studies the properties of a new class of codes introduced recently (and currently being used) in the context of digital system test compaction [12]. These codes, named X-codes after the "X" symbol used to denote unknown logic values in digital systems, address the problem of detecting errors in the presence of unknowns. Specifically, an X-code produces a hash (or signature) over a set of input bits in a way that guarantees that errors in the inputs lead to changes in the hash despite the presence of unknown inputs. In contrast with erasure-based approaches, X-codes assume that operations (*e.g.*, solution of linear equations) to calculate the values of unknown inputs are impractical or impossible. In effect, the input values are unknowable. These properties do not allow characterization in terms of Hamming distance alone, and, to the best of our knowledge, X-codes have not been studied prior to their recent introduction. The X-code may also be useful for other applications, such as communications, but our primary focus is currently on digital system designs.

For the purposes of this paper, we restrict our discussion to binary linear X-codes, although X-codes could be generalized in the future. The paper is structured as follows. The next section describes the digital system testing problem and the practical issues that give rise to X-codes. In later sections, we develop a mathematical formulation for the codes, study the relationships between different classes of X-codes, discuss a few structural elements of X-code matrices, and compare them with the superimposed codes developed for combinatorial group testing (CGT). We next describe several constructions, then conclude with a few asymptotic bounds.

2 Digital System Test Compaction

For digital systems, the voltage on a signal line is generally interpreted to be logic value 0 or 1 (except for signal lines with high impedance states). However, for many systems, some signal values cannot be uniquely determined to be in logic-0 or logic-1 state directly from the simulation

$+$	0	1	X
0	0	1	X
1	1	0	X
X	X	X	X

$\times$	0	1	X
0	0	0	0
1	0	1	X
X	0	X	X

Table 1: Addition and multiplication for 3-valued logic. Restricting operations to 0 and 1 produces GF(2), but neither addition nor multiplication forms a group over all three elements since X has no inverse in either.

model of the system. After power-up, for example, the contents of the storage elements, including memories and bistable elements such as latches or flip-flops, are unknown, and may contain either 0 or 1, depending on a range of factors. These unknown states are modeled as "X" states. Other examples of sources of X states in a digital system include floating bus lines and multiple clock domains.

Suppose that we want to test a newly fabricated integrated circuit that implements a certain digital system. The usual method is to apply input stimuli (of 0s and 1s) and to observe the circuit's response using test equipment. The response to a given test input is compared to an expected, or "golden," response obtained by performing a 3-valued (0, 1 and X) logic simulation of the system, often leading to the presence of X's in the golden response. A large body of literature studies simulation models of digital systems with X states, starting from Eichelberger's work on 3-valued logic simulation [4]. For example, if an exclusive-OR (XOR) gate has any unknown inputs, its output is also unknown. Table 1 shows the addition (XOR) and multiplication (AND) operations for a 3-valued logic system (0, 1 and X). While the two operations, when restricted to the values 0 and 1, are simply GF(2), neither operation is a group over all three elements, as X has neither an additive nor a multiplicative inverse. Similarly, distributive properties do not hold. For example,

$$0 = X \times 0 = X \times (1 + 1) \quad \neq \quad (X \times 1) + (X \times 1) = X + X = X$$

Output bits that are not X's in the golden response are compared with the corresponding bits from the actual circuit and a chip is declared to be defective (*e.g.*, due to the presence of manufacturing defects) when there is a mismatch. Generally, a digital system contains tens of millions of transistors, hundreds of thousands of flip-flops representing bits of state, and a few hundred input and output pins. Hence, it is important to compact the responses of the circuits being tested before they are transmitted over the pins for observation by the external world (*i.e.*, the tester).

Due to the importance of the test problem, substantial effort has been made to design methods for compacting (encoding) test responses, beginning with Benowitz *et al.* [1]. Early work in the area applied compaction using linear feedback shift registers [5] and multiple-input signature registers [9] to compact a sequence of test responses. Saluja and Karpovsky [14] applied standard codes to provide compaction for a single test response. As the delivery of the hash from the compactor to the testing equipment is effectively noiseless, any conventional (n, k) code (such as Hamming, BCH, or Golay) can be used to compacts n bits of response input into $m = n - k$ bits of hash output, and the code retains its conventional Hamming distance for error control in the test responses [14]. In digital system testing, the primary method is to detect faulty circuits through error

detection, but error correction can be used to diagnose and analyze failures in order to improve the manufacturing process. Although not common in current practice, fault location and diagnosis can also be used to produce a working circuit with reduced functionality, as has been done in the past at coarse granularities. As diagnosis is not typically performed for all chips, Hamming distances of 2 or 3 usually suffice in practice for digital system testing.

Unknown logic values in test responses typically have been handled by conservatively ignoring portions of a response, or through the use of logic specific to the circuit under test or to the test response. In theory, a given unknown logic value in a test response might appear as different logic values to different gates in a circuit implementing a test response compactor, but such variability is irrelevant in practice. Bistable logic elements tend to converge quickly to either 0 or 1, and unknown logic elements can be treated as erasures without significant risk [6]. Erasure-based approaches suffer from practical limitations, however.

Several methods can be considered for applying traditional coding theory to the problem of dealing with erasures in a golden test response. Test equipment might solve systems of linear equations corresponding to known patterns of erasures, or store multiple patterns for comparison. These techniques pose challenging implementation problems in practice, however, due to the implications for processing power and storage in the test equipment. For the purposes of testing, system constraints require that erasure codes on approximately a thousand elements be solved in roughly 10 nanoseconds. Alternatively, for u erasures, 2^u patterns can be compared with the output bits for each test. A code with distance of at least $u + e$ can correct u erasures and detect e errors simultaneously. Thus, with u erasures and at most e errors, we can compare against the 2^u possible outputs for matches. Current test methodologies and architectures are based on per-pin comparisons and do not readily support this technique. Also, for large values of u, it may be unreasonable to compare against 2^u patterns. Storing all of the patterns is also impractical, as tester memory is limited. Instead, the unknown input bits could be recorded, the corresponding rows in the code matrix looked up, and the resulting bit patterns tested for a match. While recording the unknown input bits may require less additional storage than recording the output bits to be ignored, the table lookup and comparison processes may take substantial time. Several variations on these approaches are also possible, trading test time with storage requirements, and are detailed in [13].

The lack of practical techniques for handling X's in golden test responses as erasures gave rise to the notion of the X-code, which allows these values to propagate logically through a compactor while guaranteeing that any errors in other parts of the golden response are detected.

3 Definitions and Abstractions

This section provides a more formal definition of X-codes. The X-code is designed to ensure that unknowable values in the response to be compacted (the input to the compactor) do not prevent errors from appearing in the hash of the response (the output of the compactor), while assuming that any outputs that depend on any unknowable input are themselves unknowable. In this paper, we limit our discussion to binary linear codes. Such codes are implemented as circuits of XOR gates and can readily be tested [11] to ensure that the compactor itself does not somehow suppress

detection of errors in other circuits. Non-linear X-codes can be represented as arbitrary 3-valued logic functions of a set of input values, but are not easily viewed as sets of codewords.

We represent a code as an $n \times m$ matrix H (with n rows and m columns), where n is the number of bits in a test response, and m is the number of bits in the resulting hash. As with conventional binary codes, the matrix entry H_{ij} is 1 if the j^{th} bit of the hash depends on the i^{th} bit of the uncompacted response, and is 0 otherwise. The hash P is calculated by multiplying the response vector V by the matrix H: $P = VH$. The j^{th} bit of the compacted response is thus obtained by XOR-ing all bits i such that $H_{ij} = 1$.

We formulate X-codes in terms of reduced matrices. Let H be a code matrix, and let S be a set of rows of H. The *reduced matrix* H_S is the matrix formed by removing (from H) all rows in S and all columns in which any row in S contains a 1. We denote by $\mathcal{X}_{x,d}$ the class of X-codes for which any two input vectors with the same hash are separated by a Hamming distance of at least d in the presence of up to x unknown input values. More formally, the class $\mathcal{X}_{x,d}$ contains all matrices H such that, for any set S of up to x rows of H, the code defined by the reduced matrix H_S has a weight (minimum Hamming distance between codewords) of at least d. Error control is then supported by the reduced matrix in the usual form of minimum Hamming distance between input vectors with the same hash values.

$$H = \left[\begin{array}{c|c|c|c|c} 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ \hline 0 & 1 & 0 & 0 & 1 \end{array}\right] \leftarrow \text{row } r \qquad H_{\{r\}} = \left[\begin{array}{ccc} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right]$$

As an example, consider the matrix on the left above. Reducing the matrix by the row r implies removing the row itself along with the two boxed columns in which the row contains a 1. The reduced matrix is shown on the right, and from inspection can be seen to define a weight 3 code (code words are 00000, 01101, 10110, and 11011). By the symmetry of the structure of H, all of the reduced matrices define codes with weight 3, and $H \in \mathcal{X}_{1,3}$.

Two points can now be made in light of the definition of X-codes. First, the reduced matrix formulation is equivalent to the use of 3-valued logic when calculating hashes. An unknowable input r value (an X) produces unknowable output values in exactly those positions (matrix columns) in which the row corresponding to r contains 1s. As errors in other input values do not influence unknowable outputs, the matrix $H_{\{r\}}$ becomes the effective code in this case. Second, the classes $\mathcal{X}_{x,2}$ are identical to the superimposed, or x-disjunct, codes developed for combinatorial group testing (CGT), a fact discussed in more detail in Section 4.

3.1 Terminology

A few definitions are useful for discussing the properties of X-codes. The terminology reflects the digital testing context in which X-codes arose, although we have adopted conventional symbols and terms from coding theory when possible. In particular, we view the problem of finding "good" X-codes as minimization of the number of outputs m for a fixed number of inputs n.

For the purposes of this paper, any code in $\mathcal{X}_{x,d}$ can be represented by an $n \times m$ matrix H; the dimensions are specific to H, but the relationship between n and m is constrained by the properties of the class $\mathcal{X}_{x,d}$. Rows in H correspond to inputs, and columns to outputs. The *compaction ratio* of H is the ratio of the number of inputs to the number of outputs, n/m. The identity matrix on n elements, I_n, is in all X-code classes, since all reduced matrices of I_n are also identity matrices and define codes with only one codeword. We say that an X-code is *non-trivial* if its compaction ratio is greater than one.

When comparing two X-code matrices, the first is *smaller* (*larger*) than the second if the first has fewer (more) columns than the second. An X-code matrix is *optimal* for a given class and a given number of rows if no smaller matrix in the class has at least as many rows. In many cases, the class and number of rows are clear from the context, and the term optimal alone suffices. Optimal X-codes always exist (possibly the identity matrices), but are not necessarily unique. Finally, we say that a matrix H is *maximal* for a given class and a given number of columns if no other code in the class with the same number of columns has more rows than H.

3.2 Matrix Properties

We now discuss several properties of X-code matrices, focusing on the similarities and differences with traditional codes. In a number of instances, we make use of the one-to-one relationship between rows in a reduced matrix H_S and rows in the matrix H being reduced (other than those in the set S). As this relationship is fairly natural, constant reference to it tends to clutter the proofs, and we deliberately omit mention of the mapping in most cases, instead denoting rows in H_S and the corresponding rows in H with the same symbols.

By definition, any check matrix H for a conventional binary linear code of weight d is in $\mathcal{X}_{0,d}$. As with traditional codes, given any matrix $H \in \mathcal{X}_{x,d}$, one can form other matrices in $\mathcal{X}_{x,d}$ by removing any number of rows from H.

Matrices can also be extended with additional rows and columns to create useful X-codes, whereas such operations are not worthwhile for most other purposes. For example, to create an X-code in $\mathcal{X}_{0,11}$ with 257 inputs, one can construct a block diagonal matrix from a 255×40 BCH check matrix [2] and a 2×2 identity matrix, as shown below. The resulting 257×42 matrix is in $\mathcal{X}_{0,11}$, but as a conventional code does nothing more than extend the number of check bits, turning a (255,215) code into a (257,215) code. The difference in utility relies on the fact that the hash bits generated by an X-code are not subject to errors, whereas all bits in a conventional code are assumed to be subject to errors and must be protected through the code.

$$\overbrace{\left[\begin{array}{cc} \left[\begin{array}{c} BCH \\ (255,215) \end{array}\right] & 0 \\ 0 & \left[\begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array}\right] \end{array}\right]}^{42} \Bigg\} 257$$

The reduced matrices of an X-code are also X-codes. In particular, given $H \in \mathcal{X}_{x,d}$ and any set of rows S in H such that $|S| \leq x$, the reduced matrix H_S is in $\mathcal{X}_{x-|S|,d}$. Similarly, any

X-code $H \in \mathcal{X}_{x,d}$ also serves as a X-code of smaller weight or for fewer unknowns: $\mathcal{X}_{x,d} \subseteq \mathcal{X}_{x-1,d}$, and $\mathcal{X}_{x,d} \subseteq \mathcal{X}_{x,d-1}$. Furthermore, using a code to handle fewer unknowns increases its weight:

Theorem 1 $\forall x, d, \mathcal{X}_{x,d} \subseteq \mathcal{X}_{x-1,d+1}$.

Proof: Let H be a matrix in $\mathcal{X}_{x,d}$ and assume that $\mathcal{X}_{x,d} \not\subset \mathcal{X}_{x-1,d+1}$. Then there exists a set S of rows of H, $|S| \leq x-1$, and a set T of rows of H_S, $|T| < d+1$, such that the rows in T sum to 0. Pick a row $r \in T$. Columns in which r contains a 1 do not appear in $H_{S\cup\{r\}}$, thus the sum of the rows corresponding to those in $T \setminus \{r\}$ is also 0. But since $|S \cup \{r\}| \leq x$ and $|T \setminus \{r\}| < d$, $H \notin \mathcal{X}_{x,d}$, a contradiction which completes the proof.

The converse relationship does not necessarily hold. Consider, for example, a code $H \in \mathcal{X}_{2,2}$, and let $H_{\{r\}} \in \mathcal{X}_{1,2}$ be the reduced matrix for some row r. No row q in $H_{\{r\}}$ can contain only 0s, nor can any row have 1s in a subset of the columns in which a second row has 1s, as otherwise $(H_{\{r\}})_{\{q\}}$ has weight 1, and $H_{\{r\}} \notin \mathcal{X}_{1,2}$. In contrast, with $G \in \mathcal{X}_{1,3}$, the rows in a reduced matrix need only be non-zero and unique, implying that at least three are required to sum to 0. Patterns that obey the constraint for $\mathcal{X}_{2,2}$ also obey the constraint for $\mathcal{X}_{1,3}$ (equivalent bit patterns are subsets of one another), but not vice-versa.

Similarly, the code on the left below is the smallest non-trivial code in $\mathcal{X}_{1,2}$. The Hamming code on the right has both fewer columns and more rows, and is optimal and maximal in $\mathcal{X}_{0,3}$, but is not in $\mathcal{X}_{1,2}$, as an unknown value in the last row hides all errors in other inputs.

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \qquad \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

These subset relationships form a partial ordering on the X-code classes. Given classes $\mathcal{X}_{x,d}$ and $\mathcal{X}_{x',d'}$ with $x > x'$, the ordering is undefined if and only if $x + d < x' + d'$.

3.3 Submatrices and Row Weights

The X-code matrices also exhibit a number of interesting properties based on submatrices and row weights. The following theorem generalizes a property of conventional codes to X-codes.

Theorem 2 *For any $H \in \mathcal{X}_{x,d}$, and for any set S of up to x rows of H, and any set T of fewer than $(d + x - |S|)$ rows of the reduced matrix H_S, there exists a set C of $|T|$ columns of H_S such that the submatrix formed by the intersection of T and C in H_S has determinant 1.*

Proof: As $x \geq |S|$ and $x + d > |S| + |T|$, we have $H \in \mathcal{X}_{|S|,|T|+1}$, and $H_S \in \mathcal{X}_{0,|T|+1}$. The rest of the proof is adapted from conventional coding theory. Consider the submatrix X of H_S formed by the rows in T (with all columns). Build the set C by starting with any non-zero column from

X and repeatedly adding columns that are linearly independent from all others already in C. If fewer than $|T|$ can be found, some non-empty subset of T sums to 0, contradicting the fact that $H_S \in \mathcal{X}_{0,|T|+1}$ and completing the proof.

This theorem makes it easy to generalize one of the results from the CGT literature for arbitrary classes of X-codes:

Corollary 1 *For any $H \in \mathcal{X}_{x,d}$ and any set S of up to $x+1$ rows of H, there exists a set C of $|S|$ columns of H such that the submatrix formed by the intersection of S and C in H is a permutation matrix (an identity matrix under row or column permutations).*

Proof: For each $r \in S$, apply Theorem 2 to find a column c_r of $H_{S\setminus\{r\}}$ in which r contains a 1. The presence of c_r in $H_{S\setminus\{r\}}$ implies that all rows in $S \setminus \{r\}$ contain 0s in c_r. The set C is defined to be all of the c_r, completing the proof.

The presence of such permutation submatrices also suffices to prove membership in $\mathcal{X}_{x,2}$ (as shown in the CGT literature):

Corollary 2 *Let H be a code matrix such that, for any set S of up to $x+1$ rows of H, there exists a set C of $|S|$ columns of H such that the submatrix formed by the intersection of S and C in H is a permutation matrix. Then $H \in \mathcal{X}_{x,2}$.*

Proof: Let S be a non-empty set of up to $x+1$ rows, and let C be a set of columns such that the submatrix lying in both S and C is a permutation matrix. Pick any $r \in S$. Clearly, the vector of weight 1 with a single 1 in the position corresponding to r is not a codeword of $H_{S\setminus\{r\}}$. However, the choices of S and r were arbitrary, thus neither H nor any reduced matrix of H (by up to x rows) has any codeword of weight 1, and $H \in \mathcal{X}_{x,2}$.

Finally, we can place a lower bound on row weight (number of 1s in a row) for small codes:

Theorem 3 *Given $H \in \mathcal{X}_{x,d}$ such that H is a smallest non-trivial code in $\mathcal{X}_{x,d}$, the weight of any row of H is at least $x+1$.*

Proof: Assume the contrary, and let row r of H have weight $q \leq x$. If r has a 1 in some column c of weight 1, row r and column c can be removed from H to form a smaller non-trivial code in $\mathcal{X}_{x,d}$, which contradicts the assumption that H is a smallest non-trivial code. Thus no column in which r has a 1 has weight 1. Form a set S of rows by starting with row r and choosing, for each column c in which r has a 1, another row q_c that has a 1 in c. Set S has cardinality of at most $q+1 \leq x+1$, but violates Corollary 1, as no column exists in which row r alone has support. Thus $H \notin \mathcal{X}_{x,d}$, a contradiction that completes the proof.

The implication of this theorem is that, when trying to construct codes from a given class, we should consider consider only rows with weight greater than x. Codes on fewer inputs and outputs can be extended trivially by including the identity matrix, and rows of weight greater than one but not greater than x serve no useful purpose.

4 Relationship to Superimposed Codes

This section describes superimposed codes and illustrates their relationship to X-codes, then discusses a handful of bounds and constructions no doubt already found in the literature on superimposed codes. Superimposed codes were introduced by Kautz and Singleton [7] in 1964 for the problem of combinatorial group testing (CGT), and have been an active area of study in both the information theory and mathematics communities since that time. A paper by D'yachkov, Macula, Jr. and Rykov [3] serves as a good starting point for exploring the rich literature in this area.

Combinatorial group testing arose from the need to screen soldiers for syphilis. The test process allowed blood samples pooled from a number of soldiers to be tested simultaneously, with a positive test result obtained whenever one or more of the samples so pooled indicated the presence of syphilis. As the fraction of infected individuals was expected to be small, and tests were expensive, codes were developed to identify infected individuals within a group without testing each member separately, *i.e.*, by only testing subgroups.

More formally, given a group of n individuals and a target maximum number x of infected individuals, the original CGT problem requires a set of tests such that the results of the tests either uniquely identify up to x infected individuals or indicate that more than x individuals are infected. If the tests must be designed in advance (to allow them to proceed simultaneously, for example), the problem is termed deterministic group testing, and the set of m predefined tests can be represented as an $n \times m$ matrix in which rows correspond to individuals and columns to group tests. The superimposed codes [7] were designed for this problem. As an example of later directions of interest, subsequent generalizations on this problem address false positives and false negatives in the test results.

We are now ready to present a formulation of superimposed codes and to demonstrate that they are equivalent to the classes $\mathcal{X}_{x,2}$. Given a code matrix H, a set S of rows of H *obscures* a row r of the reduced matrix H_S if r contains only 0s. Using this definition, we can write the definition of $\mathcal{X}_{x,2}$ as follows: a matrix H is in $\mathcal{X}_{x,2}$ iff for any set S of up to x rows of H and any other row r of H, $r \notin S$, S does not obscure r. In CGT terms, a matrix is said to be x*-disjunct* if no set of up to x rows obscures any other row. Such matrices are also called *superimposed codes*.

Theorem 4 *Let H be an $n \times m$ matrix. For any $x \leq n$, the following are equivalent:*

(a) the matrix H defines a set of m group tests that solve the original CGT problem for n individuals with no more than x infected, and

(b) $H \in \mathcal{X}_{x,2}$.

Proof: ((a) implies (b)) Assume that H solves the CGT problem for some value of x, but that $H \notin \mathcal{X}_{x,2}$. Then there exists some set S of up to x rows of H and another row r of H, $r \notin S$, such that S obscures r. Assume that all individuals corresponding to rows in S are infected. Since S obscures r, no group test can then determine whether or not r is also infected. However, the set $S \cup \{r\}$ is either a distinct set of cardinality not more than x or a failure case (when $|S| = x$), thus the inability to distinguish S from $S \cup \{r\}$ implies that H does not solve the specified CGT problem, a contradiction.

((b) implies (a)) Assume that $H \in \mathcal{X}_{x,2}$, but that H does not solve the specified CGT problem. Then for some set S of up to x individuals, there exists a second set $T \neq S$ (T can be of any size) such that infection in exactly the members of S cannot be distinguished from infection in exactly the members of T by the group tests. Assume without loss of generality that $T \not\subset S$ (if $T \subset S$, swap the two), and pick an individual $r \in T \setminus S$. As the test results are indistinguishable for S and T, no test can include r without also including some member of S. Thus the set of rows corresponding to individuals in S obscures the row corresponding to r, but $|S| \leq x$, so $H \notin \mathcal{X}_{x,2}$, a contradiction that completes the proof.

The literature on superimposed codes serves as a good source of information for the classes $\mathcal{X}_{x,2}$. In the remainder of this section, we demonstrate a few aspects of these classes of X-codes that are presumably already known to readers familiar with that literature, but may nonetheless be interesting to readers who are not.

4.1 A Few $\mathcal{X}_{x,2}$ Constructions

The first construction employs Sperner's Theorem [8, 15] to bound optimal compaction ratios in $\mathcal{X}_{1,2}$. Rather than simply reference this theorem, however, we provide a proof so as to allow us to more readily generalize the approach to bound compaction ratios for other $\mathcal{X}_{x,2}$ classes.

Let $S = \{s_1, \ldots, s_m\}$ be a set. A *chain* C of S is an ordered set $\{T_1, \ldots, T_k\}$ of subsets of S such that $T_1 \subset T_2 \subset \ldots \subset T_k$. Chain C is *maximal* if $|C| = m + 1$. A bijection can be constructed from the permutations of S to the maximal chains of S by associating a permutation $P = (p_1, \ldots, p_m)$ with the maximal chain given by $\{\emptyset, \{s_{p_1}\}, \{s_{p_1}, s_{p_2}\}, \ldots\}$. An *antichain* A of the set S is a set $\{U_1, \ldots, U_k\}$ of subsets of S such that no member of A is a subset of another member. We are now ready to prove Sperner's Theorem.

Theorem 5 (Sperner's Theorem)
For any antichain A of a set S of m elements, $|A| \leq \left(\begin{array}{c} m \\ \lfloor m/2 \rfloor \end{array} \right)$.

Proof: By the definition of an antichain, no maximal chain of S can contain more than one element of A. Let U be a member of A, and let $q = |U|$. The number of maximal chains in which U appears can be calculated by counting the permutations in which the elements of U appear in the first q positions. In particular, there are $q!$ orderings for the elements of U, and $(m-q)!$ orderings of the elements not in U, for a total of $q!(m-q)!$ maximal chains. Let p_i be the number of elements (subsets of S) of cardinality i in A. The number of maximal chains in which any element of A appears must be less than the total number of maximal chains of S:

$$\begin{aligned} \sum_{i=0}^{m} p_i \; i! \; (m-i)! &\leq m! \\ \sum_{i=0}^{m} \frac{p_i}{\left(\begin{array}{c} m \\ i \end{array} \right)} &\leq 1 \end{aligned} \tag{1}$$

But for all values $0 \leq i \leq m$, $\left(\begin{array}{c} m \\ i \end{array} \right) \leq \left(\begin{array}{c} m \\ \lfloor m/2 \rfloor \end{array} \right)$. Thus

$$\begin{aligned} \sum_{i=0}^{m} \frac{p_i}{\left(\begin{array}{c} m \\ \lfloor m/2 \rfloor \end{array} \right)} &\leq \sum_{i=0}^{m} \frac{p_i}{\left(\begin{array}{c} m \\ i \end{array} \right)} \leq 1 \\ \sum_{i=0}^{m} p_i &\leq \left(\begin{array}{c} m \\ \lfloor m/2 \rfloor \end{array} \right) \\ |A| &\leq \left(\begin{array}{c} m \\ \lfloor m/2 \rfloor \end{array} \right) \end{aligned}$$

which completes the proof.

The bound placed by Sperner's Theorem is tight, since all subsets of $\lfloor m/2 \rfloor$ elements form an antichain with cardinality equal to the bound. Equation (1) is known as the Lubell-Yamamoto-Meshalkin inequality, and is a generalization of Sperner's Theorem.

An *antichain generator G of order x* for the set S is a set $\{V_1, \ldots, V_k\}$ of subsets of S with properties defined inductively on x. In particular, an antichain generator of order 0 is simply a set of non-empty subsets of S. An antichain generator G of order x is an antichain generator of order $x-1$ such that the union of any x distinct members of G is unique and such that the unions of all combinations of x distinct members of G together form an antichain F of set S. The antichain F is said to be *generated by G for order x*, and any antichain generator of order $x > 0$ generates x distinct antichains. Observe that an antichain generator of order 1 is simply an antichain, and that the single antichain it generates is itself. Antichain generators of order x correspond to matrices in $\mathcal{X}_{x,2}$, as shown by the following lemma.

Lemma 1 *Let H be an $n \times m$ matrix. Each row r of H defines a subset V of columns of H in which the members of V are exactly those columns in which r contains a 1. Let G be the (multi-)set of subsets defined by all rows of H. Then $H \in \mathcal{X}_{x,2}$ iff G is an antichain generator of order x.*

Proof: (sufficiency) Assume that G is an antichain generator of order x for the columns of H. No row of H can contain only 0s, as G is also an antichain generator of order 0. Let S be a non-empty set of up to x rows of H, and let A be the antichain generated by G for order $|S|$. Let $V_1, \ldots V_{|S|}$ be the elements of G corresponding to the rows of S, and let $U = V_1 \cup \ldots \cup V_{|S|}$ be their union. Then $U \in A$. Pick a row $r \notin S$, and let $R \in G$ represent r in G. If $R \subset U$, we can define a second member of A as $U' = R \cup V_2 \cup \ldots \cup V_{|S|}$, which includes R in place of V_1. Clearly, $U' \subseteq U$, implying that A is not an antichain (or that G is not an antichain generator of order $|S|$, if the two are equal), a contradiction proving that $R \not\subset U$. But $R \not\subset U$ implies that S does not obscure r. Since the choice of S and r were arbitrary (with $S = \emptyset$ handled earlier), $H \in \mathcal{X}_{x,2}$.

(necessity) Assume that $H \in \mathcal{X}_{x,2}$. No row in H can contain only 0s, thus G is an antichain generator of order 0. To prove that G is an antichain generator of order x, it must be shown that for any $1 \leq k \leq x$, G is an antichain generator of order k. Let S and T be two distinct subsets of

k rows of H (they may overlap, but cannot be equal). Let $V_1, \ldots V_k$ be the elements of G corresponding to the rows in S, and let $W_1, \ldots W_k$ be the elements of G corresponding to the rows in T. Define $U_S = V_1 \cup \ldots \cup V_k$ and $U_T = W_1 \cup \ldots \cup W_k$. If $U_S \not\subset U_T$ and $U_T \not\subset U_S$, the proof is complete. Pick a row $r \in S \setminus T$, and let $V_1 \in G$ correspond to r. If $U_S \subset U_T$, we also have $V_1 \subset U_T$, and T obscures row r. But $|T| = k \leq x$, contradicting the fact that $H \in \mathcal{X}_{x,2}$. Thus $U_S \not\subset U_T$. Similarly (or by the arbitrary choice of S and T), we also have $U_T \not\subset U_S$, completing the proof.

Applying the same technique used to prove Sperner's Theorem allows us to bound the cardinality of antichain generators:

Lemma 2 (extension of Sperner's Theorem) *For any antichain generator G of order $x > 0$ for a set S of m elements, $\left(\begin{array}{c} |G| \\ x \end{array} \right) \leq \left(\begin{array}{c} m \\ \lfloor m/2 \rfloor \end{array} \right)$.*

Proof: The antichain generated by G for order x has cardinality $\left(\begin{array}{c} |G| \\ x \end{array} \right)$. Application of Sperner's Theorem thus completes the proof.

We can now place an upper bound on the number of rows in any matrix $H \in \mathcal{X}_{x,2}$:

Theorem 6 *For any $n \times m$ matrix $H \in \mathcal{X}_{x,2}$, $\left(\begin{array}{c} n \\ x \end{array} \right) \leq \left(\begin{array}{c} m \\ \lfloor m/2 \rfloor \end{array} \right)$.*

Proof: The construction of Lemma 1 gives the antichain generator G of order x corresponding to H, and $|G| = n$. Application of Lemma 2 then proves the theorem.

As the bound placed by Sperner's Theorem is tight, we can give more specific bounds for $\mathcal{X}_{1,2}$.

Corollary 3 *Let $H \in \mathcal{X}_{1,2}$ be an $n \times m$ matrix with $n > 1$. The matrix H is maximal iff $n = \left(\begin{array}{c} m \\ \lfloor m/2 \rfloor \end{array} \right)$, and is optimal iff $\left(\begin{array}{c} m-1 \\ \lfloor (m-1)/2 \rfloor \end{array} \right) < n$.*

Proof: To prove the first part of the corollary, let A be the antichain on m elements consisting of all subsets of cardinality $\lfloor m/2 \rfloor$. Then A is an antichain of order 1, and by Lemma 1 corresponds to a matrix $X \in \mathcal{X}_{1,2}$. The matrix X has $\left(\begin{array}{c} m \\ \lfloor m/2 \rfloor \end{array} \right)$ rows and m columns, thus H is maximal iff it has the same number of rows. Proving the second part of the corollary requires only the analogous antichain-based construction on $m - 1$ columns.

We now provide a constructive lower bound for $\mathcal{X}_{2,2}$.

Theorem 7 *For any optimal code in $\mathcal{X}_{2,2}$, $m \leq \lceil \log_2 n \rceil \, (\lceil \log_2 n \rceil + 1)$.*

Proof: The proof is constructive, and produces a matrix $X \in \mathcal{X}_{2,2}$ for any given number of rows n. To simplify the discussion, let $q = \lceil \log_2 n \rceil$. The matrix X has n rows and $q(q+1)$ columns. In the submatrix formed by the first q columns, assign unique binary combinations to each row. Any combination, including all 0s, can be used. A second submatrix of $q(q-1)/2$ columns is then formed by summing (XOR'ing) each pair of columns from the first submatrix. The final submatrix of $q(q+1)/2$ columns is then formed by complementing the columns of both of the first two submatrices. The total number of columns is then $q(q+1)$.

It remains to be shown that $X \in \mathcal{X}_{2,2}$. By Corollary 2, it suffices to show that for any set S of three rows of X, there exists a set C of three columns such that the submatrix formed by the intersection of S and C is a permutation matrix.

Consider the submatrix of X induced by any three rows. It suffices to show that for any row r of the three, a column exists in the submatrix with a 1 in r and 0s in the other two rows. As the three rows are distinct, there exist columns that distinguish r from each of the other rows. In particular, the first submatrix of X is filled with distinct binary patterns for each row, thus we can always find a column with one of the following two forms to differentiate r from the second row (with r as the first row): $C_1 = [1\ 0\ a]^{\mathrm{T}}$ or $C_2 = [0\ 1\ b]^{\mathrm{T}}$. If $a = 0$ for any column of the form C_1, or if $b = 1$ for any column of the form C_2 (use the column containing the bitwise complement), we have found the necessary column and are done. If not, we have either $[1\ 0\ 1]^{\mathrm{T}}$ or $[0\ 1\ 0]^{\mathrm{T}}$ in the first submatrix.

Similarly, we can always find a column with one of the following two forms to differentiate r from the third row: $C_3 = [1\ c\ 0]^{\mathrm{T}}$ or $C_4 = [0\ d\ 1]^{\mathrm{T}}$. If $c = 0$ for any column of the form C_3, or if $d = 1$ for any column of the form C_4 (again use the column containing the bitwise complement), we have found the necessary column and are done. If not, we have either $[1\ 1\ 0]^{\mathrm{T}}$ or $[0\ 0\ 1]^{\mathrm{T}}$ in the first submatrix.

Four combinations remain. If $[1\ 0\ 1]^{\mathrm{T}}$ and $[1\ 1\ 0]^{\mathrm{T}}$ appear in the first submatrix of X, the bitwise complement of their sum ($[1\ 0\ 0]^{\mathrm{T}}$) appears in the third submatrix. The other combinations are similar, and in each case show that the desired column must exist, completing the proof.

Corollary 2 can be also used to construct an optimal X-code $H \in \mathcal{X}_{x,2}$ using an NP-complete algorithm that is practically viable for small codes. For a given number of rows n, the algorithm first constructs $x+1$ columns for each combination of $x+1$ rows. The submatrix formed by each set of columns with the associated set of rows is an identity matrix, while all other values in the columns are marked as "don't care," which we shall denote by D. The value D is said to be *compatible* with any other value, whereas 0 is only compatible with 0 and D (but not 1), and 1 is only compatible with 1 and D (but not 0). Two columns are compatible if the values in each of their rows are compatible. The algorithm forms a minimal set of compatibility classes (sets of columns) such that any two columns in a given class are compatible with one another. Next, for each compatibility class, the algorithm selects a representative column of 0s and 1s that is compatible with all columns in the class. If the columns in a class all contain D in some row, either value may be chosen for that row in the representative column for the class, but picking 0 in such cases gives a probabilistic advantage for handling more unknowns. These representative columns together form an optimal matrix in $\mathcal{X}_{x,2}$ for n rows.

Unfortunately, while small codes may be feasible with the algorithm just presented, it may not scale well for two reasons. First, the number of distinct combinations of $x+1$ rows may be large, forbidding construction of the original matrix. A logical construction should suffice, however. Second, computation of a minimal set of compatibility classes is equivalent to a minimal clique cover, and is thus an NP-Complete problem unless the structure of the original matrix can somehow be exploited.

4.2 Relation to Steiner Systems

Certain types of combinatorial designs called Steiner systems [16, 17] serve as optimal codes for $\mathcal{X}_{x,2}$, and seem to be much more effective than the construction of Theorem 7. Borrowing conventions from [17], a combinatorial design t-(v,k,λ) consists of a set S of k-subsets of v elements (*i.e.*, subsets of cardinality k), such that every t-subset of the v elements appears in exactly λ members of S. When $\lambda = 1$, the design is called a Steiner system, after the author of [16], who was one of the first to study the problem. As an example, consider the incidence matrix of the 2-(9,3,1) Steiner system, as shown below (the transpose of the incidence matrix defined in [17]). In this matrix, each row represents a 3-subset ($k=3$) of the 9 outputs ($v=9$), and each pair of outputs ($t=2$) appears in exactly one row ($\lambda=1$). The matrix is in $\mathcal{X}_{2,2}$ and $\mathcal{X}_{1,4}$.

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

The use of Steiner system incidence matrices for X-codes generalizes to some degree. We can reduce the problem of finding a code $H \in \mathcal{X}_{x,2}$ (with $x > 1$) to the problem of constructing a Steiner system as follows. Assume that the rows of the matrix must have equal weight z of the form $z = xy+1$ for some $y \geq 1$, and that we can somehow identify the appropriate value of z. Let S be a set of up to x rows of H, and let r be a row in H_S. If any two rows in the code have 1s in at most y columns in common, the weight of r in H_S is at least $xy+1-|S|y \geq 1$, since reduction by each row in S removes at most y columns in which row r contains a 1. Thus no vector of weight 1 is a codeword for H_S, and $H \in \mathcal{X}_{x,2}$. We can accomplish this goal by asserting that every subset of $y+1$ columns appears together in at most one row, which constrains the number of rows. For a fixed number of outputs m, we wish to maximize the number of rows n, which represent inputs. The answer to this new problem, and thus a code in $\mathcal{X}_{x,2}$ with a maximal number of rows for m columns and row weight z, is a Steiner system: a combinatorial design in which every $(y+1)$-subset appears in exactly one $(xy+1)$-subset, or $(y+1)$-$(m,(xy+1),1)$.

Steiner systems do not exist for arbitrary values of m, and in fact are quite sparse for even small values of y and x. Furthermore, Steiner systems may not be optimal, as solutions with variable row weights may be superior. They do, however, provide some insight on the potential for X-codes, and in some cases are provably both optimal and maximal. Consider the following theorem.

Theorem 8 *The Steiner system 2-(9,3,1) is the smallest non-trivial code in $\mathcal{X}_{2,2}$, and is maximal.*

Proof: We omit the proof, which is fairly straightforward using constraints from earlier theorems.

We can conjecture a lower bound on the asymptotic behavior of the compaction ratio for maximal codes in $\mathcal{X}_{2,2}$ by assuming that the structure of Steiner systems can be approximated (to within a non-exponential factor) when they do not exist for some values of t, v, and k. For a given value of m, we would like to construct a Steiner system of the form $(y+1)-(m, 2y+1, 1)$ for whatever value of y maximizes the number of rows n. If the Steiner system exists, n is given by

$$n = \prod_{i=0}^{y} \frac{i+m-y}{i+y+1} \tag{2}$$

For any given value of m, n rises monotonically in y to its maximum value, then falls monotonically. We maximize n by finding the minimum value of y such that:

$$\begin{aligned}
\prod_{i=0}^{y} \frac{i+m-y}{i+y+1} &\geq \prod_{i=0}^{y+1} \frac{i+m-y-1}{i+y+2} \\
\frac{(2y+3)(2y+2)}{y+1} &\geq m-y-1 \\
5y+7 &\geq m \\
y &\geq \frac{m-7}{5}
\end{aligned}$$

Thus $y = \lceil (m-7)/5 \rceil$. For large values of m, we approximate as $y = m/5$ and rewrite (2) as

$$\begin{aligned}
n &= \prod_{i=0}^{m/5} \frac{i+4m/5}{i+m/5} \\
n &= 4 \ \frac{m!(m/5)!}{(4m/5)!(2m/5)!} \\
n &= \sqrt{10} \ \frac{(1/5)^{m/5}}{(4/5)^{4m/5}(2/5)^{2m/5}} \\
n &= \sqrt{10} \ \left(\frac{5}{4}\right)^{m}
\end{aligned}$$

in which the third step uses Stirling's approximation for the factorials. The normalization constant in front is inaccurate due to the approximation of y, and furthermore does not reflect the fact that few of the desired Steiner systems actually exist. Steiner systems are sparse for even small values

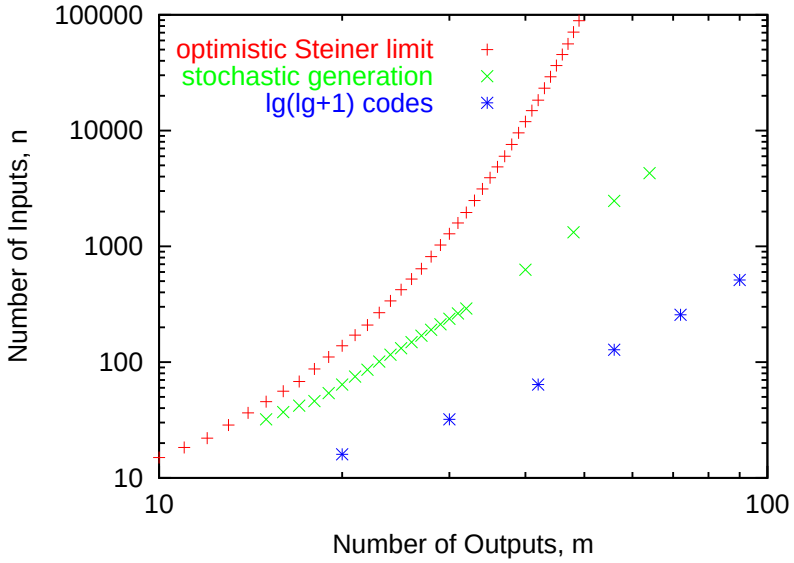

Figure 1: Bounds and constructions for $\mathcal{X}_{2,2}$. The upper line shows an optimistic bound based on Steiner systems; the middle shows the best codes generated by a simple stochastic approach; the lower shows the construction of Theorem 7. The stochastic generation used at most 7 bits per row, which becomes less effective than 9 bits per row in the Steiner bound at 23 output bits.

of y. As an example, $m \in [28, 32]$ gives $y = 5$, but the smallest value of m for which a Steiner system of the form 6-$(m, 11, 1)$ can exist (other than the trivial case of $m = 11$) is 221, and the next value is 389. Thus the maximal X-code matrixes can only approximate their structure. Figure 1 compares this bound with the construction of Theorem 7 and random codes with small fixed row weights. Tighter upper bounds can also be constructed by including the effects of the floor functions that must be applied after each multiplication in Equation 2 (the factors are all integral for Steiner systems), and linear programming using inequalities for codes of equal weight along with the Johnson bound can tighten the bound further, but are beyond the scope of this paper.

Systematic constructions do exist for certain Steiner systems, such as those of the form 2-$((x+1)^2, (x+1), 1)$ whenever $x + 1$ is a prime power. In particular, matrix columns in such a system correspond to points in a two-dimensional vector space over GF($x + 1$), and rows correspond to all one-dimensional subspaces and their cosets under addition [17].

Conjecture 1 *If for some $x > 1$ the Steiner system 2-$((x+1)^2, (x+1), 1)$ exists, it is a smallest non-trivial code in $\mathcal{X}_{x,2}$, and is maximal.*

5 The $\mathcal{X}_{1,d}$ Codes

In this section, we consider a set of general and specific constructions for codes in the $\mathcal{X}_{1,d}$ classes. One method for constructing X-codes is to start with a conventional check matrix and to invert the reduction process by extending the original matrix. Standard codes (in $\mathcal{X}_{0,d}$), for example, can be used to construct codes in $\mathcal{X}_{1,d-1}$ or $\mathcal{X}_{1,d}$ by doubling the number of columns.

Theorem 9 *Given an $n \times m$ matrix $H \in \mathcal{X}_{0,d}$, construct an $n \times 2m$ matrix X by replacing each 0 in H with the 1×2 matrix $[0\ 1]$ and each 1 in H with the matrix $[1\ 0]$. Then $X \in \mathcal{X}_{1,d-1}$, and $X \in \mathcal{X}_{1,d}$ if d is odd.*

Proof: Assume that $X \notin \mathcal{X}_{1,d}$, and let r be a row of X and S be a set of fewer than d rows of the reduced matrix $X_{\{r\}}$ such that $\sum_{s \in S} s = 0$. We now use s specifically to denote the row s in $X_{\{r\}}$, and denote by $F(s)$ (or $F(r)$) the row s (or r) in H (not X). Then, based on the construction of X, $s = F(s) + F(r)$.

If $|S|$ is even, $\sum_{s \in S} F(s) = \sum_{s \in S} s = 0$, as the $F(r)$ terms cancel. Since $|S| < d$, however, this result contradicts the assumption that $H \in \mathcal{X}_{0,d}$. If $|S|$ is odd, $\sum_{s \in S \cup \{r\}} F(s) = \sum_{s \in S} s = 0$. If $|S| < d - 1$, this fact leads to the same contradiction as the even case, completing the first part of the proof. To complete the second part of the proof, observe that $|S| < d$ implies $|S| < d - 1$ when $|S|$ and d are both odd.

While this construction provides lower bounds on the compaction ratios of codes in $\mathcal{X}_{1,d}$, the codes produced in this fashion are typically neither optimal nor maximal. Consider, for example, the $\mathcal{X}_{1,3}$ matrices generated from Hamming codes, such as the one shown below:

$$H = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \rightarrow \quad X = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \end{bmatrix}$$

One improvement to the code on the right is the addition of the row $[010101]$, which corresponds to the $[000]$ row not present in the Hamming code on the left. Even with this extension, however, the code is not maximal for six columns, as we show with a second construction specific to $\mathcal{X}_{1,3}$.

Theorem 10 *Given matrices $H_1, H_2 \in \mathcal{X}_{1,3}$, construct a third matrix X by concatenating all combinations of rows from H_1 and H_2; rows of X thus correspond to the tuples (r_1, r_2) with r_1 a row in H_1 and r_2 a row in H_2. Then $X \in \mathcal{X}_{1,3}$.*

Proof: Assume that $X \notin \mathcal{X}_{1,3}$, and let $r = (r_1, r_2)$ be a row of X and T be a set of rows of X such that $r \notin T$, $|T| \leq 2$, and the rows in T in the reduced matrix $X_{\{r\}}$ sum to 0. As $H_1 \in \mathcal{X}_{1,3}$, no row of the reduced matrix $(H_1)_{r_1}$ equals zero, nor do any two rows sum to zero. The same constraints hold for $(H_2)_{r_2}$, since $H_2 \in \mathcal{X}_{1,3}$. For the case in which $|T| = 1$, let $T = \{(t_1, t_2)\}$. The reduced form of row (t_1, t_2) must then be exactly 0, implying that $t_1 = r_1$ and $t_2 = r_2$, and contradicting the fact that (r_1, r_2) is unique in X. When $|T| = 2$, let $|T| = \{(t_1, t_2), (u_1, u_2)\}$. The fact that the reduced forms of the two rows sum to zero then implies that $t_1 = u_1$ and $t_2 = u_2$, contradicting the fact that (t_1, t_2) is unique in X and completing the proof.

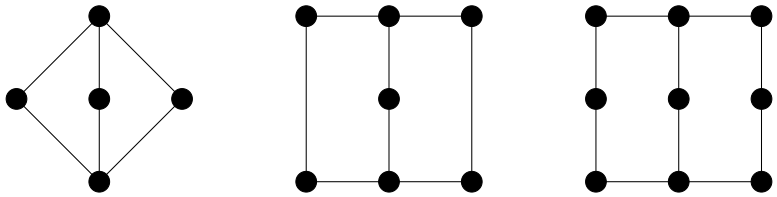

Figure 2: Graphs used to construct small codes in $\mathcal{X}_{1,d}$.

The construction of Theorem 10 can be applied repeatedly to generate matrices in $\mathcal{X}_{1,3}$ from arbitrarily long tuples of smaller matrices in $\mathcal{X}_{1,3}$. Identity matrices I_2 and I_3 work well for this purpose, and give optimal results for small numbers of outputs (we have proven to at least seven). Tuples of I_3 with zero, one, or two elements from I_2 are currently the best known codes in $\mathcal{X}_{1,3}$, *i.e.*, they provide the largest number of inputs for a given number of outputs.

The matrix constructed from the Hamming code using Theorem 9, after being extended with the row $[010101]$, is equivalent to a 3-tuple of I_2. A 2-tuple of I_3, however, allows nine inputs rather than eight, as shown below:

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

Tuples of I_3 provide the following bound:

Corollary 4 *For any optimal $n \times m$ matrix $H \in \mathcal{X}_{1,3}$, $m \leq 3\lceil \log_3 n \rceil$.*

Proof: Let $q = \lceil \log_3 n \rceil$. Construct a matrix $X \in \mathcal{X}_{1,3}$ as a q-tuple of I_3 through repeated application of Theorem 10. The matrix X has $3q$ columns and $3^q \geq n$ rows, proving the corollary.

Small non-trivial codes in any $\mathcal{X}_{1,d}$ can be constructed easily by thinking of the codes as graphs. Recall from Theorem 3 that the rows of a smallest non-trivial matrix in $\mathcal{X}_{x,d}$ must have weight of at least $x+1$, thus we consider rows with weight two for the $\mathcal{X}_{1,d}$ classes. However, a matrix in which every row has weight two is equivalent to a graph $G(V, E)$ in which each column is represented by a node and each row is represented by an edge between two nodes. The following then holds:

Theorem 11 *Given a graph $G(V, E)$, construct an $|E| \times |V|$ matrix X from G as follows. Each node $v \in V$ corresponds to one column of X, and each edge in $(u, v) \in E$ corresponds to a row of X with 1s only in the columns corresponding to u and v. Then $X \in \mathcal{X}_{1,d}$ iff G contains no cycles of length less than $d + 1$.*

Proof: (sufficiency) Assume that G has no cycles of length less than $d + 1$, and pick $T \subseteq E$ such that $2 \leq |T| \leq d$. Let the graph $G'(V, T)$ be the graph G with all edges not in T removed. Let

P be the set of vertices of degree 1 in G'. Note that if the rows in X corresponding to the edges in T are summed, all columns corresponding to points in P sum to 1. Since G has no cycles of length less than $d+1$, G' is acyclic, and $|P| \geq 2$. Similarly, since $|T| \geq 2$, $P \neq \{u, v\}$ for any $(u, v) \in T$. If $(u, v) \in T$ corresponds to an unknowable input, the reduced matrix includes all points in $P \setminus \{u, v\}$, which cannot be empty. The sum of the rows corresponding to edges in $T \setminus \{(u, v)\}$ is thus non-zero, and as the choice of T was arbitrary, $X \in \mathcal{X}_{1,d}$.

(necessity) Assume that $X \in \mathcal{X}_{1,d}$ but that a cycle of length less than $d+1$ also exists in G. Pick one edge in the cycle, and let r be the corresponding row in X. As every vertex in the cycle has exactly two edges incident on it, the rows corresponding to the remaining edges of the cycle sum to zero in the reduced matrix $X_{\{r\}}$. However, there are fewer than d such rows, which contradicts the assumption that $X \in \mathcal{X}_{1,d}$ and completes the proof.

The preceding theorem can be used to construct the smallest non-trivial codes for $d \in \{3, 4\}$ (and possibly for higher values of d as well) as follows. Construct a ring of $2d-2$ nodes, then add a bridge between diametrically opposed nodes on the ring using an extra node as a bridge, as shown in Figure 2 for $d \in \{3, 4, 5\}$. Application of Theorem 11 to the resulting graph produces small, non-trivial codes in $\mathcal{X}_{1,d}$. We make the following conjecture:

Conjecture 2 *Let $G(V, E)$ be a graph with vertices $V = \{v_1, \ldots, v_{2d-1}\}$ and edges $E = \{(v_1, v_2), (v_2, v_3), \ldots, (v_{2d-2}, v_1)\} \cup \{(v_1, v_{2d-1}), (v_{2d-1}, v_d)\}$. The $2d \times (2d-1)$ matrix corresponding to graph G is the smallest non-trivial code in $\mathcal{X}_{1,d}$.*

We have proven this conjecture for small values of d through brute force (enumerative) reasoning, but such methods do not scale well. The minor theorem below serves as an example; note that the matrix is also a tuple of I_2 and I_3, with the first and third columns forming the rows of I_2 (see Theorem 10).

Theorem 12 *The 6×5 matrix below represents the smallest non-trivial code in $\mathcal{X}_{1,3}$, and is maximal and unique:*

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

Proof: Let H be the matrix of the smallest non-trivial code in $\mathcal{X}_{1,3}$. From Theorem 3, we know that each row in H has weight of two or more. Pick a row r of H. All rows in the reduced matrix $H_{\{r\}}$ must be non-zero and unique for the code defined by $H_{\{r\}}$ to have weight 3. Thus, with m outputs, we have at most 2^{m-2} rows in H, which implies that no non-trivial code exists for $m \leq 4$. Similarly, for $m = 5$, all rows must have weights of exactly two, as the existence of a row with higher weight limits the number of rows to four or fewer. With weight limited to one or two in $H_{\{r\}}$, only seven rows are possible in H. A code with all seven possible rows, however, is not in $\mathcal{X}_{1,3}$, and one row must be eliminated, giving the matrix shown above. (All choices result in matrices isomorphic under permutations of rows and columns.)

6 Bounds

This section discusses asymptotic bounds on compaction ratios. Although we apply the techniques here only to the $\mathcal{X}_{x,3}$ classes as examples, several of the approaches are readily generalized to $\mathcal{X}_{x,d}$. We assume that the matrices of interest are both optimal and maximal, and that the rows of such matrices in $\mathcal{X}_{x,d}$ (with $x \geq 1$) have equal weight. While this assumption is clearly not valid for codes in $\mathcal{X}_{0,d}$, the process of forming reduced matrices seems to encourage more symmetric patterns, such as those necessary to meet the bound of Sperner's Theorem.

The *fractional row weight* f of an $n \times m$ matrix is the row weight divided by m, *i.e.*, the total number of 1s in the matrix divided by nm. The function $A_{x,d}(m, f)$ specifies the maximum number of rows in a matrix $H \in \mathcal{X}_{x,d}$ with m columns and fractional row weight f. We assume that all of the $A_{x,d}$ converge asymptotically (in m) to the form $O(\text{polynomial of } m)\ \ [a_{x,d}(f)]^m$, and omit the polynomial scaling factor.

We calculate an upper bound based on the recursive relationship defined by matrix reduction between classes of X-codes. Given an $n \times m$ matrix $H \in \mathcal{X}_{x,d}$, let r a row of H. The weight of r is fm. The fractional row weight f' of the reduced matrix $H_{\{r\}} \in \mathcal{X}_{x-1,d}$ limits the maximum number of rows in H:

$$A_{x,d}(f, m) \ \leq \ A_{x-1,d}((1-f)m, f')$$

In order to make the bound as tight as possible, the row r should be chosen to minimize $A_{x-1,d}(f')$. As long as the function $A_{x-1,d}$ does not have a local minimum at f, the best value that can be guaranteed is $f' = f$, as all rows may obey this equality. The bound then becomes

$$\begin{aligned} A_{x,d}(f, m) &\leq A_{x-1,d}((1-f)m, f) \\ [a_{x,d}(f)]^m &\leq [a_{x-1,d}(f)]^{(1-f)m} \\ a_{x,d}(f) &\leq [a_{x-1,d}(f)]^{(1-f)} \end{aligned}$$

Note that if a local minimum occurs at f, the bound does not apply, as a row with $f' = f$ may not exist.

The function $a_{0,3}(f)$ is readily calculated. While it is not useful for limiting codes in $\mathcal{X}_{0,3}$, it does (with given assumptions) bound the structure of the reduced matrices of codes in $\mathcal{X}_{x,3}$. The maximum number of rows of weight fm with m columns is simply

$$\begin{aligned} \left(\begin{array}{c} m \\ fm \end{array} \right) &= \frac{m!}{(fm)!\ ((1-f)m)!} \\ &= [f^f\ (1-f)^{(1-f)}]^{-m} \end{aligned}$$

by application of Stirling's approximation, and leaving out the polynomial factors. Thus $a_{0,3}(f) = 1/[f^f\ (1-f)^{(1-f)}]$, which is—perhaps not surprisingly—two raised to the entropy of f. Calculation of these bounds for several $\mathcal{X}_{x,3}$ classes appears in Figure 3.

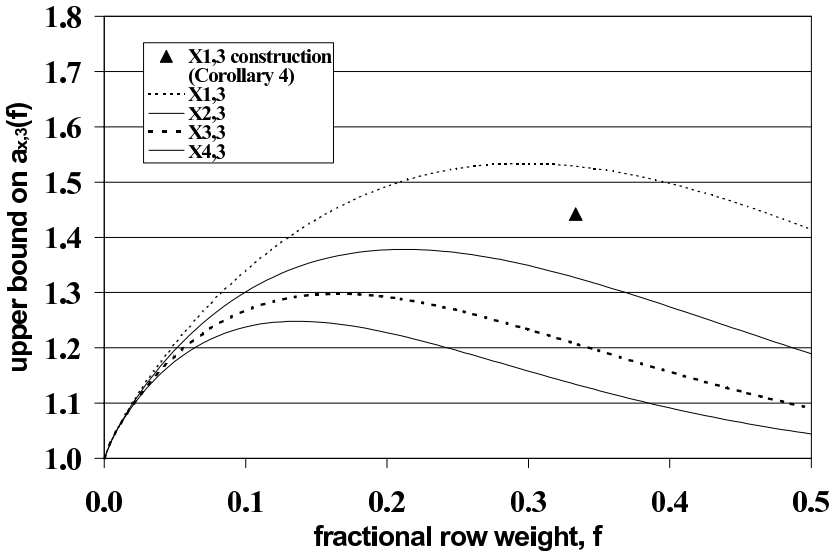

Figure 3: Upper bounds for a few $\mathcal{X}_{x,3}$. The construction is that of Corollary 4.

Acknowledgements

This work was funded in part by National Science Foundation grant ACI-99-84492-CAREER. The content of the information does not necessarily reflect the position or the policy of that organization. The authors wish to thank the following individuals for their advice, encouragement, and support: Bella Bose, Ralf Koetter, Kee Sup Kim, Ed McCluskey, and Janak Patel. Ralf Koetter suggested the qualifier "unknowable" as a way of distinguishing X-codes from erasure codes. We thank also the anonymous reviewers of the short version of this report published in the ISIT proceedings [10], who brought the relationship to superimposed codes to our attention.

References

[1] N. Benowitz, D. F. Calhoun, G. E. Alderson, J. E. Bauer, and C. T. Joeckel. An Advanced Fault Isolation System for Digital Logic. *IEEE Trans. Comput.*, C-24(5):489–97, May 1975.

[2] R. C. Bose and D. K. Ray-Chaudhuri. On a Class of Error Correction Binary Group Codes. *Inf. Control*, 3:68–79, 1960.

[3] A. G. D'yachkov, A. J. Macula, Jr., and V. V. Rykov. New Constructions of Superimposed Codes. *IEEE Transactions on Information Theory*, 46(1):284–90, January 2000.

[4] E. B. Eichelberger. Hazard Detection in Combinational and Sequential Switching Circuits. *IBM J. Res. Develop.*, 9(2):90–9, 1965.

[5] R. A. Frohwerk. Signature Analysis: A New Digital Field Service Method. *HP Journal*, pages 2–8, May 1977.

[6] G. S. Greenstein and J. H. Patel. E-PROOFS: A CMOS Bridging Fault Simulator. In *IEEE Intl. Conf. on Comp.-Aided Design (ICCAD)*, pages 268–71, 1992.

[7] W. Kautz and R. Singleton. Nonrandom Binary Superimposed Codes. *IEEE Transactions on Information Theory*, 10(4):363–77, October 1964.

[8] D. E. Knuth. *The Art of Computer Programming*, volume 3. Addison-Wesley, Reading, Massachusetts, 1973.

[9] G. Konemann, J. Mucha, and G. Zwiehoff. Built-in Logic Block Observation Technique. In *Proc. IEEE Test Conf.*, pages 37–41, October 1979.

[10] S. S. Lumetta and S. Mitra. X-Codes: Error Control with Unknowable Inputs. In *Proceedings of the International Symposium on Information Theory*, page 102, Yokohama, Japan, June 2003.

[11] E. J. McCluskey. Design Techniques for Testable Embedded Error Checkers. *IEEE Computer*, 23(7):84–8, 1990. Special Issue on Fault-Tolerant Systems.

[12] S. Mitra and K. S. Kim. X-Compact: An Efficient Response Compaction Technique for Test Cost Reduction. In *Proceedings of the International Test Conference*, pages 311–20, October 2002.

[13] J. Patel, S. S. Lumetta, and S. M. Reddy. Application of Saluja-Karpovsky Compactors to Test Responses with Many Unknowns. Technical report, University of Illinois, Center for Reliable and High-Performance Computing, 2002. In preparation.

[14] K. K. Saluja and M. Karpovsky. Testing Computer Hardware through Data Compression in Space and Time. In *Proc. Intl. Test Conf.*, pages 83–9, 1983.

[15] E. Sperner. Ein Satz über Untermengen einer endlichen Menge. *Math. Zeitschrift*, 27:544–8, 1928.

[16] J. Steiner. Combinatorische Aufgabe. *Journal für die reine und angewandte Mathematik*, 45:181–2, 1853.

[17] V. D. Tonchev. *Combinatorial Configurations*, volume 40 of *Pitman Monographs and Surveys in Pure and Applied Mathematics*. Longman Scientific and Technical, 1988.