Data verification and reconciliation with generalized error-control codes

Mark G. Karpovsky[*] Lev B. Levitin Ari Trachtenberg

Reliable Computing Lab
Boston University

Abstract

We consider the problem of data reconciliation, which we model as two physically separated multi-sets of data that must be reconciled with minimum communication. Under this model, we show that the problem of reconciliation is equivalent to a variant of the graph coloring problem and provide consequent upper and lower bounds on the communication complexity of reconciliation. More interestingly, we show by means of an explicit construction that the problem of reconciliation is equivalent to the problem of finding good error-correcting codes, provided the set of transformations has two general properties. We show analogous results for the problem of multi-set verification, in which we wish to determine whether two multi-sets are equal using minimum communication.

[*]The work of this author was supported by the NSF under grant MIP9630096

1 Introduction

The problem of reconciling data is inherent to applications that require consistency among distributed information, including diverse examples such as gossip protocols for distributing networked data [1, 2], resource discovery [3], synchronization of mobile data [4],and reconciliation of sequences of symbols from a given alphabet, such as nucleotide sequences in DNA or amino acids sequences in proteins [5]. In each of these examples, the system needs to determine the differences between data stored in physically separate locations, thereby reconciling it.

From the perspectives of scalability and performance, it is important that reconciliations occur with minimum communication, measured both by the number of transmitted bits and by the number of rounds of communication. When data are represented by sets, as can be reasonable modeled for the examples cited above, this problem is known as the *set reconciliation* problem [6, 7]. The *data reconciliation* problem is a natural generalization in which data is represented by multi-sets rather than sets.

This paper examines the data reconciliation problem within a generalized framework in which differences between multi-sets correspond to evaluations of arbitrary "error" functions. We limit communication between reconciling hosts to a single message.Under such conditions, we show that the problem of data reconciliation is equivalent to a variation of the problem of graph coloring: second-order coloring or distance-2 coloring. A second-order coloring of a graph assigns colors to vertices in such a way that any two nodes separated by a path of length at most two are colored differently. Applying well known results from graph coloring, we then provide lower and upper bounds on the amount of information that must be sent between two hosts for this type of general reconciliation.

In many practical cases, it is not necessary to reconcile two multi-sets, but merely to determine whether they are in fact the same. For example, two hosts might want to verify the correctness of a previous reconciliation, or to check whether reconciliation is needed in the first place. Such a determination can often be made with substantially less communication than a full-scale reconciliation. In this context, we consider the problem of data verification: verifying that two multi-sets are the same, subject to a given range of possible differences. Again we show that data verification is equivalent to graph coloring and error detection. We also provide both lower and upper bounds on the amount of information that must be exchanged for data verification.

The main contribution of this work is a constructive connection between generalized error-correcting codes and data reconciliation on the one hand, and between generalized error-detecting codes and data verification on the other hand. In the case of bijective and commutative errors, any data reconciliation scheme generates at least one error-correcting code, and any error-correcting code can be transformed into a corresponding data reconciliation scheme. The same correspondence exists between data verification and error-detecting codes.

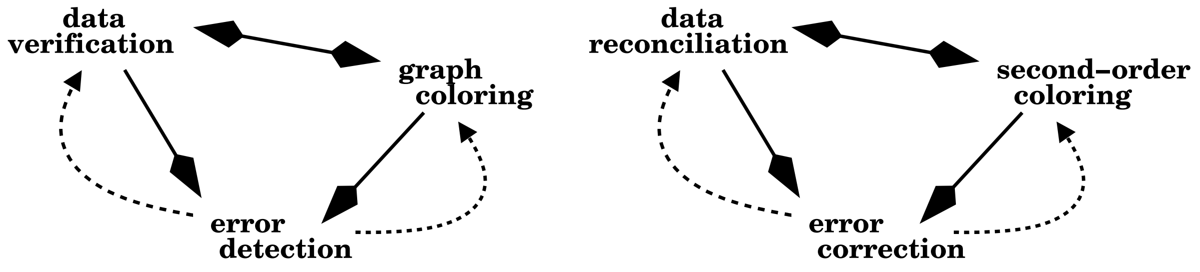

Figure 1: The various connections proven in this paper. Solid lines indicate an unconstrained reducibility between problems, and dotted lines indicate reducibility only under bijective and commutative errors.

Collecting these two equivalences, we thus have that under bijective and commutative errors, data comparison is equivalent to error-control.

However, the situation is quite different in the general case where errors are not bijective and/or not commutative. Any data comparison scheme (*i.e.* verification or reconciliation) still generates an error-control code. However, some error-control codes cannot be transformed into a corresponding data comparison scheme. In particular, the redundancy of an error-control code for such errors may be smaller than the number of different signals required for the corresponding data comparison. Thus, in general, data comparison requires more communication than error control, suggesting an intrinsic difference between the two problems. Figure 1 graphically displays the connections we describe in this paper.

1.1 Related work

The problem of reconciliation has been studied extensively from many different perspectives in the literature. We can broadly characterize the different techniques based on their model of the differences between two reconciling hosts.

One model involves synchronizing two discrete random variables with some known joint probability distribution using a minimum communication complexity. Witsenhausen [8] followed by Alon and Orlitsky [9] show a connection between such random variable reconciliation and graph coloring, giving results analogous to those of Section 3.1 and 4.1.

Another model involves two hosts reconciling files (or strings) that differ by a bounded number of insertions, deletions, or modifications (collectively: "edits"). The problem of efficient reconciliation under these circumstances, also known as the edit-distance problem [10], has been extensively studied [11, 12] because of its connections to important fields such as file synchronization and pattern recognition. Levenšteĭn [13] pioneered work in this area by developing error-correcting codes capable of correcting precisely these types of errors. Recently [14] Levenšteĭn also examined the problem of reconstructing a sequence from several copies distorted with these types of errors.

In contrast, this paper examines a model that slightly different from those mentioned above. In our model, data on two reconciling hosts is represented by *multi-sets* that differ by a very general class of differences. Being stored as multi-sets, the data on the two hosts is inherently unindexed, meaning that only the content of the individual data items, and not their relative position, matters; however the other models presented can be encompassed by this model.

2 Background

Definition 1 *A* proper coloring *of a graph G with set of vertices V and edges E is an assignment of colors to each vertex in such a way that the vertices of any edge $e \in E$ are colored differently.*

A proper coloring using at most k colors will be called a *k-coloring* of the graph. The *chromatic number* of a graph, denoted $\chi(G)$, is the minimum integer k for which there exists an k-coloring of G.

Definition 2 *A* second-order coloring *of a graph is a a proper coloring of a graph with the extra property that no two neighbors of any vertex have the same color.*

A second-order coloring of G is also a proper coloring of the square of the graph, which is the graph G^2 obtained from G by additionally connecting with an edge each pair of vertices that are of distance two apart. The minimum number of colors needed to second-order color a graph is the *second order chromatic number* of the graph, denoted $\chi_2(G)$.

Error detection and correction. Consider the module $\mathbb{Z}_q^n$ consisting of all n-dimensional vectors over the ring $\mathbb{Z}_q$. A q-ary code of length n is simply a subset of the elements of this module.

Definition 3 *An* error set *for $\mathbb{Z}_q^n$ is a set $\mathcal{E} = \{e_0, e_1, e_2, \ldots e_{|\mathcal{E}|}\}$ whose elements are functions $e_i : \mathbb{Z}_q^n \longrightarrow \mathbb{Z}_q^n$, one of which is the identity function $e_0(x) = x$. If the functions $e_i \in E$ are all bijections and their inverses e_i^{-1} are also in $\mathcal{E}$, then we shall call this set* bijective. *If the functions commute with each other, so that $e_i(e_j(x)) = e_j(e_i(x))$ $\forall x \in \mathbb{Z}_q^n, \forall e_i, e_j \in \mathcal{E}$, we shall call this set* commutative.

Example 1 (Classical errors) The classical problem of error detection and correction involves distortions of up to t q-ary symbols in a vector of length n. The error functions corresponding to all such distortions of a vector are given by translations in $\mathbb{Z}_q^n$. More formally, this error set is given by

$$\mathcal{E}_{\mathsf{class}}^t = \{e_v(x) = x + v \quad | \quad v \in \mathbb{Z}_q^n \text{ and } \|v\| \leq t\},$$

where $\|v\|$ denotes the Hamming weight of a vector v, and $+$ denotes component-wise addition mod q. Clearly, the set $\mathcal{E}_{\mathsf{class}}^t$ is both commutative and bijective.

Definition 4 *The $\mathcal{E}$-*vicinity *for a given vector $x \in \mathbb{Z}_q^n$ and an error set $\mathcal{E}$ is denoted by $\mathcal{E}(x)$ and defined to be $\mathcal{E}(x) = \{e(x) \mid e \in \mathcal{E}\} \cup \{z \in \mathbb{Z}_q^n \mid e(z) = x, e \in \mathcal{E}\}$. In general, the $\mathcal{E}^k$-*vicinity*, denoted $\mathcal{E}^k(x)$, is defined to be*

$$\mathcal{E}^k(x) = \bigcup_{y \in \mathcal{E}^{k-1}(x)} \mathcal{E}(y),$$

where $\mathcal{E}^1(x)$ in this notation denotes the set $\mathcal{E}(x)$.

A code that detects an error set $\mathcal{E}$ must be able to properly distinguish corruptions from elements of the code.

Definition 5 *A code $C \in \mathbb{Z}_q^n$* detects *the error set $\mathcal{E}$ if*

$$c_i \notin \mathcal{E}(c_j) \quad \forall c_i \neq c_j \in C.$$

Definition 6 *A code $C \in \mathbb{Z}_q^n$* corrects *the error set $\mathcal{E}$ if*

$$\mathcal{E}(c_i) \cap \mathcal{E}(c_j) = \emptyset \quad \forall c_i \neq c_j \in C.$$

Set and multi-set reconciliation and verification. The traditional formalization of the set reconciliation problem is as follows [6, 7]: given a pair of hosts A and B, each with a set of length-b bit-strings (denoted S_A and S_B respectively) and no *a priori* knowledge of the other host's set, how can each host determine the mutual difference of the two sets with a minimal amount of communication.

In general, we may consider data represented as multi-sets whose elements are chosen from a finite, universal set U. A multi-set is defined to be a set whose element multiplicities are significant. Thus, every multi-set M whose elements are taken from U may be associated uniquely with a *characteristic vector* $v(M)$ of length $|U|$ whose i-th component is j if and only if the i-th element of U occurs j times in M, for some canonical ordering of the elements of U.

We shall generally assume in this paper that the multiplicity of any element is bounded by $q-1$ meaning that $v(M) \in \mathbb{Z}_q^{|U|}$. We further limit ourselves to the case where only one of the two hosts needs to determine the multi-set held by the other host, based on information transmitted in one message.

We thus formally define a data reconciliation function, which acts upon the characteristic vectors v_A and v_B of two host multi-sets.

Definition 7 *The function $\sigma : \mathbb{Z}_q^n \longrightarrow \Sigma$ is a* one-way data reconciliation function *for an error set $\mathcal{E}$ if there exists a recovery function $R : (\Sigma \times \mathbb{Z}_q^n) \longrightarrow \mathbb{Z}_q^n$ reconciling multi-sets that differ by one of the errors in $\mathcal{E}$. More precisely, the recovery function must have the property that*

$$\forall v_A, v_B \subseteq \mathbb{Z}_q^n, e \in \mathcal{E}, \qquad v_A = e(v_B) \quad or \quad e(v_A) = v_B \implies R(\sigma(v_A), v_B) = v_A.$$

The transmission size *of such a reconciliation function is the the number of signals $|\Sigma|$ that need to be transmitted for reconciliation.*

We are also interested in the problem of data verification, due to its connections to set reconciliation and a variety of independent applications such as off-line testing [15] and signature analysis [16].In all these cases, two hosts seek to confirm that they have the same multi-set, subject to a known list of possible error functions.

We define verification functions in terms of characteristic vectors.

Definition 8 *A function $\alpha : \mathbb{Z}_q^n \longrightarrow \Sigma$ is a* one-way data verification function *for an error set $\mathcal{E}$ if there exists a decision function $D : (\Sigma \times \mathbb{Z}_q^n) \longrightarrow \{0,1\}$ with the property that*

$$\begin{aligned}&\forall v_A, v_B \subseteq \mathbb{Z}_q^n, \ \ e \in \mathcal{E},\\ &\qquad [v_A = e(v_B) \quad or \quad e(v_A) = v_B] \quad and \quad D(\alpha(v_A), v_B) = 1 \quad \Longleftrightarrow \quad (v_A = v_B).\end{aligned}$$

The transmission size *of such a verification function is the number of signals $|\Sigma|$ that need to be transmitted for verification.*

Example 2 The one-way set verification function for the error set given in Example 1 allows two hosts to verify that their sets are either equal or else differ by at least $t+1$ entries. When $t = 2^b$ this is a test of precise set equality.

3 Data verification, coloring, and error-detection

The data verification problem can be reformulated formally as follows. Consider two hosts A and B with multi-sets M_A and M_B respectively. The goal of verification is to determine whether $M_A = M_B$, subject to the sole *a priori* assumption that the multi-set M_B is a distortion of the multi-set M_A via some error in the set $\mathcal{E}$ (*i.e.* $v(M_A) = e(v(M_B))$ or $e(v(M_A)) = v(M_B)$ for some $e \in \mathcal{E}$). Assuming A and B know nothing about each other's multi-set beyond this assumption, the data verification problem is to determine the minimum amount of information A should send to B so that B can decide whether or not $M_A = M_B$.

3.1 Graph coloring

To develop a connection between data verification and graph coloring, consider a natural graph structure corresponding to a given error set $\mathcal{E}$.

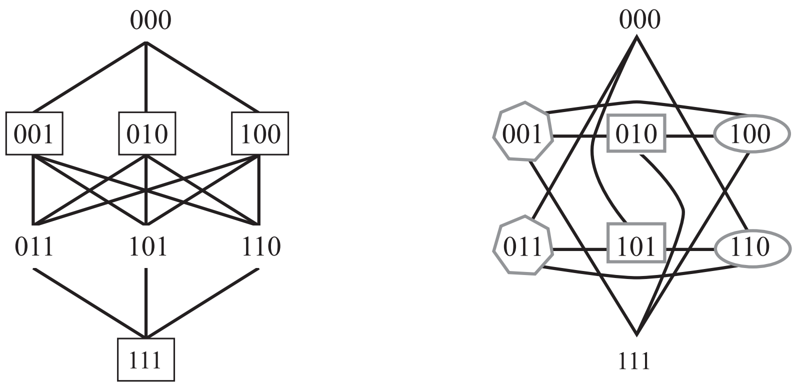

Figure 2: The characteristic graph of the error set of odd-weight translations (left) and even-weight translations (right) of length three binary vectors. The left graph can be two-colored, but the rightmost graph requires at least four colors. A sample coloring is depicted, with different outline regions denoting different colors.

Definition 9 *The* characteristic graph *of an error set $\mathcal{E}$ is the undirected graph $G_{\mathcal{E}} = (V, E)$ whose vertices are characteristic vectors of multi-sets $M \subseteq \mathbb{Z}_q^n$. Any two vertices $v_1, v_2 \in V$ are connected by an edge in this graph iff there exists a non-identity error $e \in \mathcal{E}$ such that $e(v_1) = v_2$ or $v_1 = e(v_2)$.*

Theorem 1 *Any proper coloring of $G_{\mathcal{E}}$ generates a one-way data verification function $\alpha(\cdot)$ for the error set $\mathcal{E}$. Conversely, any verification function $\alpha(\cdot)$ yields a proper coloring of $G_{\mathcal{E}}$. The minimum transmission size required for any such verification is precisely the chromatic number $\chi(G_{\mathcal{E}})$.*

The following example demonstrates a case where verification is particularly simple.

Example 3 Consider an error set consisting of all odd-weight translations errors in $\mathbb{Z}_2^n$. The characteristic graph of this error set is given in Figure 2 for parameter values $q = 2$ and $n = 3$ (*i.e.* binary vectors of length 3). Clearly this graph can be two-colored, indicating that set verification can be done with the transmission of one bit. However, if we simply change the error set to consist of all even-weight translations, we need two bits of transmission. In general, odd-weight translation errors will require only 1 bit for verification, whereas even-weight translation will require $n - 1$ bits.

Corollary 1 *The minimum transmission size W_V for a one-way data verification function over an error set $\mathcal{E}$ satisfies the inequalities*

$$W_V \;\leq\; \max_{v \in \mathbb{Z}_q^n} |\mathcal{E}(v)| \;\leq\; 2|\mathcal{E}|. \tag{1}$$

Though a non-optimal proper coloring satisfying the upper bound in (1) can be generated in linear time, practical use of such techniques is severely limited by the fact that the size of the characteristic graph grows exponentially in the size of underlying multi-sets being verified. For example, verifying over a universal set of 32-bit integers would require coloring a graph with $2^{2^{32}}$ vertices. For certain classes of errors, a more practical approach is based on error-detecting codes.

3.2 Error detection

Recall that a level set of a function is a set of points at which the function returns the same value.

Theorem 2 *Any one-way data verification function $\alpha(\cdot)$ for an error set $\mathcal{E}$ generates codes in $\mathbb{Z}_q^n$ which detect $\mathcal{E}$. Conversely, any error-detecting code $\mathbb{C} \subseteq \mathbb{Z}_q^n$ is the level set of a one-way data verification function. Therefore, each monochromatic set of vertices in a proper coloring of $G_{\mathcal{E}}$ is also an error-detecting code for $\mathcal{E}$.*

The following two corollaries of Theorem 2 sharpen the connection between verification and error-detection.

Corollary 2 *Any one-way data verification function $\alpha(\cdot)$ over $\mathbb{Z}_q^n$ with transmission size τ corresponds to an error-detecting code with at least $\frac{q^n}{\tau}$ codewords.*

Corollary 3 *For any error set $\mathcal{E}$ with characteristic graph $G_{\mathcal{E}}$ there exists an error-detecting code $\mathbb{C}$ with number of codewords*

$$|\mathbb{C}| \geq \frac{q^n}{\chi(G_{\mathcal{E}})} \geq \frac{q^n}{\max_{v \in \mathbb{Z}_q^n} |\mathcal{E}(v)|} \geq \frac{q^n}{2|E|}.$$

We note that Corollary 3 generalizes the well-known Gilbert-Varshamov bound to our general class of errors.

The connection between verification and error detection established by Theorem 2 is unidirectional: given a verification function one immediately obtains at least one error-detecting code; however, an error-detecting code does not, in general, generate a verification function, unless the error set $\mathcal{E}$ is both bijective and commutative.

Theorem 3 *Any non-extendible[1] code $\mathbb{C} \subseteq \mathbb{Z}_q^n$ that detects a bijective and commutative error set $\mathcal{E}$ also generates a one-way data verification function $\alpha(\cdot)$ with transmission size at most $\max_{c \in \mathbb{C}} |\mathcal{E}(c)|$.*

Theorem 3 shows that data verification and error detection are equivalent for bijective and commutative error sets $\mathcal{E}$. However, in general this is not the case, as the following counterexample demonstrates.

Counterexample 1 Consider the length 3, binary, linear code consisting of four vectors: $\mathbb{C} = \{000, 010, 101, 111\}$. This code detects the error set $\mathcal{E}$ consisting of:

- an identity error $e_0(x) = x$.
- a translation error $e_1(x) = x \oplus 100$, where $\oplus$ denotes binary XOR.
- two permutation errors $e_2(x_1x_2x_3) = x_2x_3x_1$ and $e_3(x_1x_2x_3) = x_3x_1x_2$

The characteristic graph of $\mathcal{E}$ is depicted in Figure 3. This error set is bijective, but not commutative.

The verification function constructed in Theorem 3 takes on two values: $\alpha(c_i) = e_0$ and $\alpha(e_1(c_i)) = e_1$ for $c_i \in \mathbb{C}$. Verification will thus fail on vectors $v_A = 001$ and $v_B = 100$, which differ by error e_3, because it will decide that they are equal (*i.e.* $\alpha(v_A) = \alpha(v_B) = e_1$). Furthermore, the graph in Figure 3 has chromatic number 3, meaning that no one-way data verification function will match the 1-bit of redundancy needed by the error-detecting code $\mathbb{C}$.

[1] A code is extendible if codewords can be added to it without affecting its error-detecting/correcting capability.

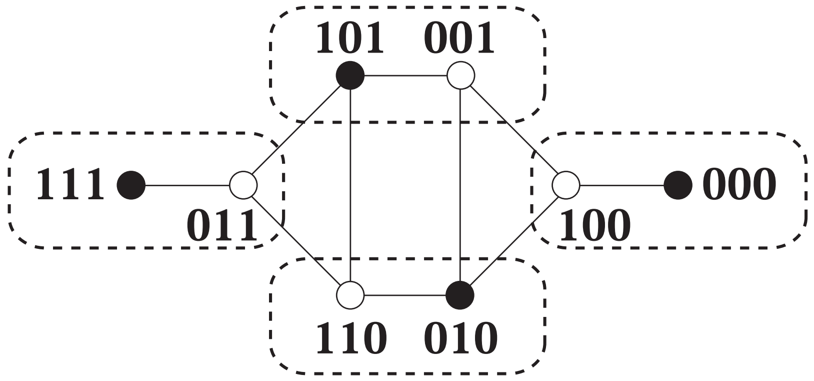

Figure 3: The characteristic graph for the error set in Counterexample 1. Solid circles denote codewords of the corresponding error-detecting code.

4 Data reconciliation, coloring, and error-correction

The problem of data reconciliation is to determine the minimum amount of information host A should send to host B so that B can determine the multi-set M_A held by A. As before, the only *a priori* assumption is that the set M_B held by B is a distortion of M_A via some error in $e \in \mathcal{E}$.

4.1 Graph coloring

Theorem 4 *Any second-order coloring of $G_{\mathcal{E}}$ generates a one-way data reconciliation function $\sigma(\cdot)$ for the error set $\mathcal{E}$. Conversely, any such reconciliation function $\sigma(\cdot)$ yields a second-order coloring of $G_{\mathcal{E}}$. The minimum transmission size required for reconciliation is precisely the second-order chromatic number $\chi_2(G_{\mathcal{E}})$.*

Corollary 4 *The minimum transmission size W_R for a one-way data reconciliation function over an error set $\mathcal{E}$ satisfies the inequalities*

$$\max_{v \in \mathbb{Z}_q^n} |\mathcal{E}(v)| \;\leq\; W_R \;\leq\; \max_{v \in \mathbb{Z}_q^n} |\mathcal{E}^2(v)|. \tag{2}$$

We note that lower bound for data reconciliation is identical to the upper bound for data verification, suggesting that for some error sets the hardest data verification problems are the easiest data reconciliation problems.

4.2 Error correction

Theorem 5 *Any one-way data reconciliation function $\sigma(\cdot)$ for an error set $\mathcal{E}$ generates codes in $\mathbb{Z}_q^n$ which correct $\mathcal{E}$. Each monochromatic set of vertices in a second-order coloring of $G_{\mathcal{E}}$ is a code that corrects $\mathcal{E}$.*

Corollary 5 *Any one-way data reconciliation function $\sigma(\cdot)$ over $\mathbb{Z}_q^n$ with transmission size τ corresponds to an error-correcting code with at least $\frac{q^n}{\tau}$ codewords.*

Corollary 6 *For any error set $\mathcal{E}$ with characteristic graph $G_{\mathcal{E}}$ there exists an error-correcting code $\mathbb{C}$ with number of codewords*

$$\frac{q^n}{\max_{v \in \mathbb{Z}_q^n} |\mathcal{E}^2(v)|} \;\leq\; \frac{q^n}{\chi_2(G_{\mathcal{E}})} \;\leq\; |\mathbb{C}| \;\leq\; \frac{q^n}{\max_{v \in \mathbb{Z}_q^n} |\mathcal{E}(v)|}.$$

By Theorem 2, any code that *detects* error set $\mathcal{E}$ is a monochromatic set of vertices in a proper coloring of $G_{\mathcal{E}}$. In contrast with this, not every code that *corrects* $\mathcal{E}$ is a monochromatic set in a second-order coloring of $G_{\mathcal{E}}$.

Theorem 6 *Any non-extendible code $\mathbb{C} \subseteq \mathbb{Z}_q^n$ that corrects a bijective and commutative error set $\mathcal{E}$ generates a one-way data reconciliation function $\sigma(\cdot)$ with transmission size at most $\max_{c \in \mathbb{C}} |\mathcal{E}^2(c)|$.*

The proof for Theorem 6 provides the following protocol for performing a reconciliation using a given code.

Protocol 1 ***(Data reconciliation using a code $\mathbb{C}$)***

Given hosts A and B with characteristic vectors v_A and v_B respectively, B can be reconciled with A as follows:

1. *A sends to B the value $\sigma(v_A) = e_A$ where $e_A(c_A) = v_A$ for some codeword $c_A \in \mathbb{C}$.*
2. *B calculates $e_A^{-1}(v_B)$.*
3. *B finds the unique $e \in \mathcal{E}$ with the property that $e_A^{-1}(v_B) = e(c)$ for some $c \in \mathbb{C}$*
4. *B determines A's characteristic vector as $v_A = e^{-1}(v_B)$.*

In the case of classical errors, one can understand Theorem 6 in terms of the covering radius of a code [17], defined to be the minimum value ρ for which balls of radius ρ will completely cover $\mathbb{Z}_q^n$.

Corollary 7 *For every non-extendible code $\mathbb{C}$ of length n, covering radius ρ, and minimum distance $2t+1$, there corresponds a one-way data reconciliation and verification function for $\mathcal{E}^t_{class}$ (from Example 1) with transmission size*

$$\sum_{i=0}^{\rho} \binom{n}{i} (q-1)^i.$$

Theorems 5 and 6 are constructive in that they provide precise instructions for how to construct a reconciliation function from a code and *vice versa*. The next example illustrates these theorems.

Example 4 (Polynomial Interpolation) Consider the protocol given in [7] for traditional set reconciliation. This protocol associates with each set $S \subseteq \mathbb{Z}_2^b$ a characteristic polynomial

$$\chi_S(x) = \prod_{s \in S} (x - s).$$

In order to reconcile, host A sends to the other host m evaluations $\chi_{S_A}(s_i)$ at prescribed points $s_i \in \mathbb{Z}_2^b$; the value m is an assumed bound on the size of the mutual difference between the sets of both hosts. By comparing χ_{S_A} to χ_{S_B}, host B is able to determine the mutual difference of the two sets.

Applying Theorem 6 we see that given sample points s_i, one code corresponding to this reconciliation is given by

$$\mathbb{C} = \{v(S) \in \mathbb{Z}_2^{2^b} \mid \chi_S(s_i) = 0 \ \forall s_i\}.$$

In other words, $\mathbb{C}$ is the cyclic code corresponding to the ideal generated by $\prod_{s_i}(x - s_i)$. When s_i are the appropriate powers of a primitive n-th root of unity, this is precisely the well-known Bose-Chaudhuri-Hochquenghem (BCH) code.

4.3 Non-commutative and non-bijective errors

Theorems 3 and 6 only provide a reduction from data comparison to error-control codes when the error set is both bijective and commutative. However, several useful error sets, such as permutations, image transformations, and insertions are either non-commutative or non-bijective.

Counterexample 2 (Uni-directional errors) Consider the case where one host is a server to which entries are added at various times, and another host is a client trying to stay up to date with the server. Reconciling such data can be modeled with the use of unidirectional errors, in which the only permitted corruptions involve changing 0's in a source vector to 1's in a target vector or vice versa. This error set is clearly non-bijective. For the case $n = 4$ and $q = 2$, the linear code $\mathbb{C} = \{0000, 0011, 1100, 1111\}$ corrects one unidirectional error using 2 bits of redundancy. On the other hand, the characteristic graph for these errors can only be second-order colored with 8 colors, meaning that reconciliation requires at least 3 bits of communication.

5 Applications

Various natural applications exist for the general class of errors modeled in this work. Problems such as set reconciliation, file synchronization, page-error correction, and client-server updates, each involve reconciliation over a different error set. We describe one of these problems.

5.1 Page errors

Consider the model of page errors, in which case errors are assumed to occur only in the same region. This could happen, for example, in the case of comparing outputs of two processors. If each processor outputs data in a certain region of the ambient space, then a processor failure on one processor would result in set differences in only the corresponding region [18].

Consider, for sake of example, that two hosts each have subsets of $\mathbb{Z}_{16}$, with page regions defined every four elements; thus the first page contains elements $\{1, 2, 3, 4\}$, the next page contains $\{5, 6, 7, 8\}$, *etc.* The error set $\mathcal{E}_{\text{page}}$ for this model thus contains all functions that corrupt a single page. For example, a corruption of the first page by a toggling the existence of set elements 1 and 3 is given by

$$e_3(x) = x \oplus 0000000000000101$$

where x is the characteristic vector of the corrupted set. Since $|\mathcal{E}(x)| = 61$ in this case, Corollary 1 implies that one-way set verification requires at most 6 bits of communication. One-way set reconciliation, on the other hand, requires $6 \le W_R(\mathcal{E}_{\text{page}}) \le \log(|\mathcal{E}^2(X)|) = \log(1411) < 11$ bits of communication. Using Reed-Solomon codes of length 4, we can attain $W_V(\mathcal{E}_{\text{page}}) \le 4$ bits and $W_R(\mathcal{E}_{\text{page}}) \le 8$ bits of communication.

In general, if sets are chosen as subsets of $\mathbb{Z}_2^b$ and each page contains p elements any t of which might be corrupted, then the amount of communication needed for reconciliation when $\frac{t}{2^b} \to 0$ and $\frac{p}{b} \to 0$ is given by

$$\log_2 W_R(\mathcal{E}_{\text{page}}) \le \log_2 \left[\sum_{i=0}^{2t} (2^p - 1)^i \binom{\frac{2^b}{p}}{i} \right] \approx 2tb \qquad (3)$$

bits, whereas a classical function correcting any pt errors would require roughly ptb bits of communication. The bound in Equation 3 corresponds to the bound in [19] based on using Reed-Solomon codes for reconciliation of these types of errors.

6 Conclusions

In this work we have studied the problem of reconciling remote data with a minimum amount of communication. We have demonstrated connections between data reconciliation, error-correcting codes, and graph coloring over a general error set. In particular, we have described in Section 4 how to transform an arbitrary code that corrects a general class of commutative and bijective errors into an algorithm for data reconciliation, and *vice versa*; similarly in Section 3 we have shown how such an error-detecting code can be used to perform data verification. These transformations are particularly useful because a wide variety of codes have already been presented in the literature for a number of different error sets.

Using the connections discovered in this paper, we have also presented the following bounds on the number of signals that need to be transmitted for data verification ($W_V(\mathcal{E})$) and data reconciliation ($W_R(\mathcal{E})$) for an arbitrary set of corruptions $\mathcal{E}$ that form a characteristic graph $G_{\mathcal{E}}$:

$$\chi(G_{\mathcal{E}}) \leq W_V(\mathcal{E}) \leq \max_x |\mathcal{E}(x)| \quad (4)$$

$$\max_x |\mathcal{E}(x)| \leq \chi_2(G_{\mathcal{E}}) = W_R(\mathcal{E}) \leq \max_x |\mathcal{E}^2(x)|. \quad (5)$$

Both upper bounds (4) and (5) are constructive in that we have described an explicit means of attaining them for a given error set. As mentioned in Section 1.1, the lower bounds have already appeared in various forms in the literature.

Finally, we have presented a number of examples throughout the work and in Section 5, thereby demonstrating the applicability of this work to such diverse areas as testing, file synchronization, and client-server network updates.

Acknowledgments

The authors wish to thank Alon Orlitsky, Krishnamurthy Viswanathan, and Junan Zhang for their interest and thoughtful discussions. One of the authors would also like to thank Yaron Minsky for stimulating discussions and Alexander Vardy for encouragement.

References

[1] R. van Renesse, Y. Minsky, and M. Hayden, "A gossip-style failure detection service," in *Middleware '98: IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing*, Nigel Davies, Kerry Raymond, and Jochen Seitz, Eds. 1998, pp. 55–70, Springer Verlag.

[2] M. Hayden and K. Birman, "Probabilistic broadcast," Tech. Rep., Cornell University, 1996.

[3] M. Harchol-Balter, T. Leighton, and D. Lewin, "Resource discovery in distributed networks," in *18th Annual ACM-SIGACT/SIGOPS Symposium on Principles of Distributed Computing*, Atlanta, GA, May 1999.

[4] A. Trachtenberg and D. Starobinski, "Towards global synchronization," *Large Scale Networks workshop*, March 2001, http://ana.lcs.mit.edu/ sollins/LSN-Workshop/papers/.

[5] R. Durbin, S. Eddy, A. Krogh, and G. Mitchéson, *Biological sequence analysis*, Cambridge university press, 1998.

[6] Yaron Minsky, Ari Trachtenberg, and Richard Zippel, "Set reconciliation with nearly optimal communication complexity," Tech. Rep. TR2000-1796,TR2000-1813, Cornell University, 2000.

[7] Yaron Minsky, Ari Trachtenberg, and Richard Zippel, "Set reconciliation with nearly optimal communication complexity," in *International Symposium on Information Theory*, 2001, to appear.

[8] H.S. Witsenhausen, "The zero-error side information problem and chromatic numbers," *IEEE Trans. on Information Theory*, vol. 22, no. 5, September 1976.

[9] N. Alon and A. Orlitsky, "Source coding and graphs entropies," *IEEE Transactions on Information Theory*, vol. 42, no. 5, pp. 1329–1339, September 1996.

[10] Alon Orlitsky, "Interactive communication: Balanced distributions, correlated files, and average-case complexity.," in *Proceedings of the 32nd Annual Symposium on Foundations of Computer Science*, 1991, pp. 228–238.

[11] G. Cormode, M. Paterson, S.C. Ṣahinhalp, and U. Vishkin, "Communication complexity of document exchange," *ACM-SIAM Symposium on Discrete Algorithms*, January 2000.

[12] T. Schwarz, R.W. Bowdidge, and W.A. Burkhard, "Low cost comparisons of file copies," *Proceedings of the International Conference on Distributed Computing Systems*, pp. 196–202, 1990.

[13] V.I. Levenšteĭn, "Binary codes capable of correcting spurious insertions and deletions of ones," *Problems of Information Transmission*, vol. 1, no. 1, pp. 8–17, 1965.

[14] V.I. Levenšteĭn, "Efficient reconstruction of sequences," *IEEE Trans. on Inf. Theory*, vol. 47, pp. 2–22, January 2001.

[15] D.P. Siewiorek and R.S. Swarz, *Reliable Computer Systems: Design and Evaluation*, Digital Press, 1992.

[16] M. G. Karpovsky and P. Nagvajara, "Design of self-diagnostic boards by signature analysis," *IEEE Trans. on Industrial Electronics*, pp. 241–246, May 1989.

[17] G.Cohen, I.Honkala, S.Litsyn, and A.Lobstein, *Covering Codes*, Elsevier, 1997.

[18] M. G. Karpovsky, T. Roziner, and C. Moraga, "Error detection in multiprocessor systems and array processors," *IEEE Trans. on Computers*, vol. 44, no. 3, pp. 383–394, March 1995.

[19] K.A.S. Abdel-Ghaffar and A.E. Abbadi, "An optimal strategy for comparing file copies," *IEEE Transactions on Parallel and Distributed Systems*, vol. 5, no. 1, pp. 87–93, January 1994.