Set Reconciliation with Nearly Optimal Communication Complexity

Yaron Minsky,* Ari Trachtenberg,† and Richard Zippel‡

September 21, 2000

Abstract

We consider the problem of efficiently reconciling two similar sets held by different hosts while minimizing the communication complexity. This type of problem arises naturally from gossip protocols used for the distribution of information. We describe an approach to set reconciliation based on the encoding of sets as polynomials. The resulting protocols exhibit tractable computational complexity and nearly optimal communication complexity. Also, these protocols can be adapted to work over a broadcast channel, allowing many clients to reconcile with one host based on a single broadcast, even if each client is missing a different subset.

1 Introduction

Gossip protocols, also known as epidemic algorithms, spread information through a network of hosts by random contacts between pairs of hosts. Through many such uncoordinated exchanges, information is spread throughout the system. Gossip protocols, while not a new idea [1, 2], have recently become the subject of increasing interest as a building block for reliable and scalable distributed systems.

The information disseminated by a gossip protocol usually consists of a set of distinct entries, each entry comprising a discrete piece of information about a system. Examples of information disseminated by gossip protocols include: addresses of participating hosts [3, 4, 5, 6]; locations of resources [7]; bibliographic data [3]; and broadcast messages [2, 3, 8, 9]. When a pair of hosts exchange information, they must reconcile their respective data sets, so that each ends up knowing the other's information. What makes this reconciliation difficult is that the hosts do not know *a priori* which data elements need to be transmitted.

We formalize the problem of reconciling two hosts' data sets as follows: given a pair of hosts A and B, each with a set of length-b bit-strings, how can each host determine the union of the two sets with a minimal amount of communication—both with respect to the number of exchanges between the two hosts and with respect to the number of bits of information exchanged. We call

*Supported in part by ARPA/RADC grant F30602-96-1-0317, AFOSR grant F49620-00-1-0198, Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory, Air Force Material Command, USAF, under agreement number F30602-99-1-0533, National Science Foundation Grant 9703470, and a grant from Intel Corporation. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of these organizations or the U.S. Government.

†A. Trachtenberg is with the Dept. of Electrical and Computer Engineering, Boston University, Email: trachten@bu.edu

‡R. Zippel is with the School of Computer Sciences, The Interdisciplinary Center, Herzliya, Email: rz@idc.ac.il

this the *set reconciliation* problem. Set reconciliation has applications outside of gossip protocols. In particular, it can be applied to any situation where unordered databases need to be reconciled. So, for example, set reconciliation could be used to improve the efficiency of reconciling distributed file-systems and databases.

This paper presents two surprisingly elegant and efficient set reconciliation protocols. Instead of dealing with sets directly, these protocols encode each set as a polynomial whose roots are the elements of the set. The advantage of this approach is that some operations on sets, in particular set difference, can be computed more efficiently from the polynomial encoding.

The communication complexity of these set reconciliation protocols is independent of the sizes of the hosts' sets, and instead depends only on the size of the difference between the two sets. Moreover, under certain conditions, set reconciliation can be achieved non-interactively, with just a single message. Thus, a host A could broadcast a kb-bit message, and every host B_i whose set differs from A's set by at most k bit-strings (each of length b) could recover the bit-strings it is missing. This works even if each host B_i is missing a different set of bit-strings, so that the total number of distinct bit-strings recovered is much larger than k.

Section 2 presents a protocol for reconciling two hosts whose sets bear a subset relationship; that is, one host's set is a subset of the other host's set. Section 3 generalizes this result to the general case where no subset relationship is assumed. Section 4 presents information-theoretic bounds on set reconciliation, and shows that the communications complexity of the protocols of Sections 2 and 3 are near optimal. Section 5 discusses related work, and Section 6 discusses directions for future research. Finally, the appendices describe some of the computational intricacies in these protocols and present some implementation data.

2 Subset Reconciliation

Consider a pair of hosts A and B that each have a set of length-b bit-strings, denoted S_A and S_B respectively. Denote the differences between the two hosts by $\Delta_A = S_A \setminus S_B$ and $\Delta_B = S_B \setminus S_A$. Let m_A and m_B the size of Δ_A and Δ_B respectively, and let $m = m_A + m_B$. The subset reconciliation problem is the special case of set reconciliation where $S_B \subseteq S_A$, *i.e.* where $m = m_A$ and $m_B = 0$.

One case of the subset reconciliation problem has a straightforward solution—when only one of A's bit-strings is not known to B, that is, when $m_A = 1$. In this case, Protocol 1 reconciles the two sets with a single b-bit message. The key to Protocol 1 is that the two hosts can recover the parity sum (bitwise exclusive or) of the difference set Δ_A from the parity sum of S_A and the parity sum of S_B. Since $|\Delta_A| = 1$ this parity sum is in fact the missing bit-string.

Protocol 1 Subset Reconciliation when $m_A = 1$

---

1. Host A computes parity_A, the parity sum of its bit-strings, and sends it to B
2. Host B computes parity_B, the parity sum of its bit-strings.
3. Host B computes the parity sum of parity_A and parity_B, which is precisely the missing bit-string.

2.1 Characteristic Polynomials

Protocol 1 is limited to the case where $m = m_A = 1$ by the fact that the parity sum does not carry enough information to recover multiple elements of a set. To generalize Protocol 1, we need a generalization of the parity-sum that provides sufficient information to recover more than one missing bit-string. The generalization we will use is the *characteristic polynomial* $\chi_S(Z)$ of a set $S = \{x_1, x_2, \ldots, x_n\}$, which we define to be the following univariate polynomial.

$$\begin{aligned} \chi_S(Z) &= (Z - x_1)(Z - x_2)(Z - x_3) \cdots (Z - x_n) \\ &= Z^n - \sigma_1(S) Z^{n-1} + \cdots + (-1)^n \sigma_n(S). \end{aligned} \tag{1}$$

The coefficients $\sigma_i(S)$ of the characteristic polynomial are known as the *elementary symmetric polynomials* of S. The i-th elementary symmetric polynomial of a set S is the sum of products of all i element subsets of S. Thus, if a set $S = \{x_1, x_2, x_3, \ldots, x_m\}$ then

$$\begin{aligned} \sigma_1(S) &= x_1 + x_2 + + \cdots + x_m, \\ \sigma_2(S) &= x_1 x_2 + x_1 x_3 + \cdots + x_{m-1} x_m, \\ \sigma_3(S) &= x_1 x_2 x_3 + x_1 x_2 x_4 + \cdots + x_{m-2} x_{m-1} x_m, \\ &\vdots \\ \sigma_m(S) &= x_1 x_2 \cdots x_m. \end{aligned}$$

Note that the zeros of $\chi_S(Z)$ are precisely the elements of S. Thus, the elements of S can be recovered by factoring $\chi_S(Z)$. Also note that $\chi_S(Z)$ is necessarily monic, *i.e.* its leading coefficient is 1.

To use the characteristic polynomial in place of the parity sum of Protocol 1, host B must be able to determine the coefficients of $\chi_{\Delta_A}(Z)$ given a small amount of information provided by A. Theorem 1 shows how the coefficients of $\chi_{\Delta_A}(Z)$ can be reconstructed from only the high-order coefficients of $\chi_{S_A}(Z)$ and $\chi_{S_B}(Z)$. Thus, once A transmits the required coefficients of $\chi_{S_A}(Z)$ to B, the coefficients of $\chi_{\Delta_A}(Z)$ can be reconstructed by B.

Theorem 1 *The coefficients of $\chi_{\Delta_A}(Z)$ can be reconstructed from the coefficients of the $m_A + 1$ highest-degree terms of $\chi_{S_A}(Z)$ and $\chi_{S_B}(Z)$. In particular, we have the following convolutional relationship between the coefficients of $\chi_{S_A}(Z)$, $\chi_{S_B}(Z)$ and $\chi_{\Delta_A}(Z)$:*

$$\sigma_k(\Delta_A) = \sigma_k(S_A) - \sigma_k(S_B) - \left[\sum_{0<i<k} \sigma_i(\Delta_A) \sigma_{k-i}(S_B) \right] \tag{2}$$

Proof: Since $S_A = \Delta_A \cup S_B$, the characteristic polynomial of S_A can be written as the product:

$$\chi_{S_A}(Z) = \chi_{\Delta_A}(Z) \chi_{S_B}(Z)$$

Equating the coefficients on both sides of the equals sign gives:

$$\begin{aligned} \sigma_1(S_A) &= \sigma_1(\Delta_A) + \sigma_1(S_B) \\ \sigma_2(S_A) &= \sigma_2(\Delta_A) + \sigma_1(\Delta_A)\sigma_1(S_B) + \sigma_2(S_B) \\ &\vdots \\ \sigma_k(S_A) &= \sigma_k(\Delta_A) + \sum_{0<i<k} \sigma_i(\Delta_A)\sigma_{k-i}(S_B) \\ &\quad + \sigma_k(S_B). \end{aligned} \tag{3}$$

Equation (2) follows from a simple rearrangement of Equation (4) and can be applied iteratively to construct the first k elementary symmetric polynomials of Δ_A from the first k elementary symmetric polynomials of S_A and S_B. These elementary symmetric polynomials are the high-order coefficients of $\chi_{\Delta_A}(Z)$. ∎

To use characteristic polynomials for set reconciliation, we need to map length-b bit-strings onto elements of some field. To eliminate growth in the size of the numbers with which we compute, we use a finite field, which we denote by $\mathbb{F}_q$. There are two cases of interest: the first is where q is prime and $\mathbb{F}_q$ is simply the integers modulo q. In this case, each bit-string is interpreted as a binary integer less than q. By Bertrand's Postulate [10, p. 343] there is always at least one prime number between 2^b and 2^{b+1}, so elements of $\mathbb{F}_q$ can be represented using $\lg q < b+1$ bits. The second case is where $q = 2^b$. Elements of $\mathbb{F}_q$ are then isomorphic to polynomials $c_{b-1}\alpha^{b-1} + c_{b-2}\alpha^{b-2} + c_{b-3}\alpha^{b-3} + \ldots + c_0$ in α, where α is the zero of an irreducible polynomial of degree b over $\mathbb{F}_2$. In this case, each bit-string is represented using exactly $\lg q = b$ bits.[1]

Using a field of order 2^b is more efficient in terms of the number of bits required, since no overhead is needed to transmit a length-b bit-string. On the other hand, using a prime-order field may require up to one extra bit per transmission, but is computationally more efficient on most hardware. For the rest of this paper we will interpret all bit-strings as elements of $\mathbb{F}_q$, without specifying a choice of q.

2.2 The Protocol

Protocol 2, which was first described in [11], provides an efficient solution to the subset reconciliation problem. Note that Protocol 2 only sends m_A coefficients, and not m_A+1, because the characteristic polynomial is necessarily monic and so the leading coefficient is known *a priori*. Note also that Protocol 2 assumes that m_A, the number of missing bit-strings, is known. Since $S_B \subseteq S_A$, the value of m_A is simply the difference $|S_A| - |S_B|$ and can be determined with a single b-bit message.

Protocol 2 Subset Reconciliation when $m_A \geq 1$

Assuming $S_B \subseteq S_A$, and host A knows m_A, hosts A and B can reconcile their data sets as follows:

1. Host A computes the first m_A elementary symmetric polynomials of the elements of its set S_A and sends them to host B.
2. Host B computes the first m_A elementary symmetric polynomials of its set S_B.
3. Using Theorem 1, host B computes the elementary symmetric polynomials of $\Delta_A = S_A \setminus S_B$ and constructs the corresponding characteristic polynomial $\chi_{\Delta_A}(Z)$. The elements of Δ_A are precisely the zeros of $\chi_{\Delta_A}(Z)$.

The following example demonstrates Protocol 2 concretely.

Example 1 Consider the set $S_A = \{1, 2, 3, 4, 5, 6\}$ and its subset $S_B = \{2, 4, 6\}$ stored as 3-bit integers at hosts A and B respectively. We treat the members of S_A and S_B as elements of $\mathbb{F}_{11}$.

- Host B sends $|S_B| = 3$ to host A.

[1] Note that when working over $\mathbb{F}_{2^m}$, $\sigma_1(S)$ is the parity sum discussed at the beginning of this section.

- Host A sends to host B the $m_A = |S_A| - |S_B| = 3$ elementary symmetric polynomial values:

$$\begin{aligned}
\sigma_1(S_A) &= 1+2+3+4+5+6 = 10,\\
\sigma_2(S_A) &= 1\cdot 2 + 1\cdot 3 + \cdots + 5\cdot 6 = 10,\\
\sigma_3(S_A) &= 1\cdot 2\cdot 3 + 1\cdot 2\cdot 4 + \cdots + 4\cdot 5\cdot 6 = 9
\end{aligned}$$

- Host B then computes its own elementary symmetric polynomial values:

$$\begin{aligned}
\sigma_1(S_B) &= 2+4+6 = 1,\\
\sigma_2(S_B) &= 2\cdot 4 + 2\cdot 6 + 4\cdot 6 = 0,\\
\sigma_3(S_B) &= 2\cdot 4\cdot 6 = 4
\end{aligned}$$

- Host B uses Equation (2) to compute the elementary symmetric polynomial values of $\Delta_A = S_A \setminus S_B$:

$$\begin{aligned}
\sigma_1(\Delta_A) &= \sigma_1(S_A) - \sigma_1(S_B) = 9,\\
\sigma_2(\Delta_A) &= \sigma_2(S_A) - \sigma_2(S_B) - \sigma_1(\Delta_A)\sigma_1(S_B) = 1,\\
\sigma_3(\Delta_A) &= \sigma_3(S_A) - \sigma_3(S_B) - \sigma_1(\Delta_A)\sigma_2(S_B)\\
&\qquad - \sigma_2(\Delta_A)\sigma_1(S_B) = 4
\end{aligned}$$

- Using the equality in Equation (1), host B reconstructs the characteristic polynomial

$$\chi_{\Delta_A}(Z) = Z^3 - 9Z^2 + 1Z - 4.$$

  The polynomial $\chi_{\Delta_A}(Z)$ is factored as $(Z-1)(Z-3)(Z-5)$ and its zeros are precisely the elements of $\Delta_A = \{1, 3, 5\}$.

2.3 Analysis

Protocol 2 requires two messages: one to determine the size m_A, and the other to transmit the m_A required coefficients. If an upper bound on m_A is known, then the protocol requires only a single message, whose length depends on the quality of the upper bound.

Including the cost of determining m_A, the protocol needs to transmit only $\lceil m_A \lg(q) \rceil + b$ bits. As noted earlier, we can either pick q to be a prime, in which case $b \leq \lg(q) \leq b+1$, or we can pick q to be 2^b, in which case $\lg(q) = b$. Choosing $q = 2^b$ gives the following communication bound.

Theorem 2 *Protocol 2 reconciles sets S_A and S_B using $b \cdot (m_A + 1)$ bits of communication.*

The computational complexity of Protocol 2 is quite tractable. There are two bottlenecks in the calculations: computation of the elementary symmetric polynomials and finding the zeros of the characteristic polynomial. The computation of the elementary symmetric polynomials can be amortized over insertions into host A's set, since

$$\sigma_i(S_A \cup \{x\}) = \sigma_i(S_A) + x \cdot \sigma_{i-1}(S_A). \tag{4}$$

Thus, to maintain the values of m elementary symmetric polynomials, host A needs only $O(m)$ time per insertion for an overall running time of $O(m|S_A|)$.

The problem of finding zeros of a polynomial over $\mathbb{F}_q$ is well studied [12, 13, 14, 15]. Appendix A describes a practical method for factoring a square-free polynomial of degree m in expected time $O(m^3 \lg q)$; more sophisticated algorithms [16] can bring this asymptotic time down to $O(m^{1.82} \lg q)$, but their practical benefits are not clear.

3 Set Reconciliation

Recall that the subset reconciliation algorithm described in Section 2 works by first recovering $\chi_{\Delta_A}(Z)$, the characteristic polynomial of the set of missing bit-strings, and then determining the roots of that polynomial in order to recover the elements of the difference set Δ_A. The coefficients of $\chi_{\Delta_A}(Z)$ are recovered from the coefficients of the characteristic polynomials of S_A and S_B respectively.

The approach used in Section 2 does not apply directly to set reconciliation for two reasons: first, the technique for recovering the coefficients of the characteristic polynomial requires that S_B be a subset of S_A; second, the algorithm relies on the fact that it is easy to determine m, the number of missing elements. Neither of these assumptions hold in the case of set reconciliation.

To deal with these problems, we adopt an approach based on sampling and rational function interpolation for recovering the required characteristic polynomials. The following section presents an overview of our approach.

3.1 Overview

The key to our approach to set reconciliation is the observation that

$$\frac{\chi_{S_A}(Z)}{\chi_{S_B}(Z)} = \frac{\chi_{\Delta_A}(Z)}{\chi_{\Delta_B}(Z)}$$

This holds because factors corresponding to elements common to both S_A and S_B cancel out in the division. Thus, although the degrees of $\chi_{S_A}(Z)$ and $\chi_{S_B}(Z)$ may be very large, the degrees of the numerator and denominator of the (reduced) rational function are much smaller, m_A and m_B respectively.

The broad outline of our approach consists of three basic steps:

1. Hosts A and B evaluate $\chi_{S_A}(Z)$ and $\chi_{S_B}(Z)$ respectively at the same $\overline{m}$ sample points, where $\overline{m}$ is an upper bound on m.

2. The sampled values are combined to compute the value of $\chi_{S_A}(Z)/\chi_{S_B}(Z)$, at each of the sample points. These values are interpolated to recover the coefficients of the reduced rational function $\chi_{\Delta_A}(Z)/\chi_{\Delta_B}(Z)$.

3. By factoring $\chi_{\Delta_A}(Z)$ and $\chi_{\Delta_B}(Z)$, the elements of Δ_A and Δ_B are recovered.

Sections 3.2 and 3.3 discuss rational function interpolation and the selection of sample points in more detail. A concrete set reconciliation protocol is then described in Section 3.4. Finally Section 3.5 discusses how set reconciliation can be solved without an *a priori* bound $\overline{m}$ on m.

3.2 Rational Function Interpolation

The problem of determining a rational function with that takes on prescribed values is called the *rational interpolation problem.* Most of the research in this area has focused on rational functions with floating point coefficients, where accuracy and numerical stability are important issues. There are two assumptions that distinguish the rational interpolation problems studied in this paper. First, the coefficient domain of our rational functions is a finite field where there are no problems of accuracy or numerical stability. Second, we know *a priori* that the desired rational functions are monic.

Let $f(Z)$ be a rational function in one variable over the field K:

$$f(Z) = \frac{p_0 Z^m + p_1 Z^{m-1} + \cdots + p_m}{q_0 Z^n + q_1 Z^{n-1} + \cdots + q_n} = \frac{P(Z)}{Q(Z)},$$

where the $p_i, q_j \in K$. If $P(Z)$ and $Q(Z)$ are relatively prime (*i.e.* they have no common factors not in K) then we say that $f(Z)$ is *reduced*. We say that $f(Z)$ is a *monic rational function* if $p_0 = q_0 = 1$. We define the *degrees* of $f(Z)$ to be (m, n). Two rational functions P_1/Q_1 and P_2/Q_2 are said to be *equivalent* if $P_1 Q_2 = P_2 Q_1$. That is, if the two rational functions are reduced to the same rational function by canceling common factors between the numerator and denominator.

A set of pairs $(k_i, f_i) \in K^2$, where the k_i are distinct, is called a *support set* for an interpolation problem. We say that a function $f(Z)$ *satisfies* the support set V if $f(k_i) = f_i$ for all $(k_i, f_i) \in V$.

We consider the problem of finding a monic rational function $f(Z)$ that satisfies a support set V of size $\overline{m}$ such that the sum of the degrees of the numerator and denominator of $f(Z)$ is less than or equal to $\overline{m}$, and the difference between the degrees of the numerator and of the denominator is d. Two issues need to be addressed: (1) the existence and uniqueness of a solution to the problem, and (2) an efficient algorithm to reconstruct a rational function from a support set.

For our use in set reconciliation, the support set is constructed from values of an existing rational function $\chi_{\Delta_A}(Z)/\chi_{\Delta_B}(Z)$, for which d, the difference between the degree of the numerator and denominator, is $|\Delta_A| - |\Delta_B| = |S_A| - |S_B|$. Thus, existence of a satisfying rational function is guaranteed. The following proposition addresses the uniqueness issue. It is an adaptation of a standard theorem on rational interpolation (see [17], Proposition 2.2.1.4).

Theorem 3 *Let V be a support set with $\overline{m}$ elements over a field K. Assume there exist two monic rational functions that satisfy the support set and that the sum of the degree of the numerator and denominator is no more than $\overline{m}$. Then the two rational functions are equivalent.*

Proof: Denote the two monic rational functions, by $P_1(Z)/Q_1(Z)$ and $P_2(Z)/Q_2(Z)$. Since both rational functions satisfy the support set, we have

$$\frac{P_1(k_i)}{Q_1(k_i)} = \frac{P_2(k_i)}{Q_2(k_i)},$$

for every k_i in the support set V. Clearing the fractions, we see that the polynomial $P(Z) = P_1(Z)Q_2(Z) - P_2(Z)Q_1(Z)$ must vanish at all k_i in the support set. Since all of the polynomials are monic, the degree of $P(Z)$ is no greater than $\overline{m} - 1$. Since $P(Z)$ vanishes at $\overline{m}$ points, it must be identically zero. ∎

Thus, rational interpolation is unique up to the equivalence relation between rational functions.

We now present an algorithm for solving the interpolation problem described above. Assume we have a support set

$$V = \{(h_1, f_1), \ldots, (h_{\overline{m}}, f_{\overline{m}})\}$$

Assume that there exists a monic reduced rational function $f(Z)$ of degrees (m_A, m_B) such that $m_A + m_B \leq \overline{m}$ and $m_A - m_B = d$. Our goal is to recover $f(Z)$ from V, $\overline{m}$ and d.

We can bound m_A and m_B as follows:

$$m_A \leq \lfloor (\overline{m} + d)/2 \rfloor \stackrel{def}{=} \overline{m}_A$$

$$m_B \leq \lfloor (\overline{m} - d)/2 \rfloor \stackrel{def}{=} \overline{m}_B$$

Note that $\overline{m}_A - m_A = \overline{m}_B - m_B$. Thus, there exists a monic rational function $P(Z)/Q(Z)$ of degrees $(\overline{m}_A, \overline{m}_B)$ which satisfies V — in particular, $f(Z)$ with the numerator and denominator multiplied by $Z^{\overline{m}_A - m_A}$. We can write $P(Z)$ and $Q(Z)$ as follows:

$$\begin{aligned} P(Z) &= Z^{\overline{m}_A} + p_1 Z^{\overline{m}_A - 1} + \cdots + p_{\overline{m}_A}, \\ Q(Z) &= Z^{\overline{m}_B} + q_1 Z^{\overline{m}_B - 1} + \cdots + q_{\overline{m}_B}. \end{aligned}$$

Each pair $(k_i, f_i) \in V$ gives rise to a linear relation between the coefficients of P and Q:

$$k_i^{\overline{m}_A} + p_1 k_i^{\overline{m}_A - 1} + \cdots + p_{\overline{m}_A} = f_i \cdot (k_i^{\overline{m}_B} + q_1 k_i^{\overline{m}_B - 1} + \cdots + q_{\overline{m}_B}).$$

We may combine $\overline{m}$ of these relations to form a *generalized Vandermonde* system of equations as shown in (5). We denote this system of equations by $\mathcal{S}(\overline{m}_A, \overline{m}_B; V)$. Being a solution of Equation (5)

$$\begin{pmatrix} k_1^{\overline{m}_A - 1} & \cdots & k_1 & 1 & -f_1 k_1^{\overline{m}_B - 1} & \cdots & -f_1 k_1 & -f_1 \\ k_2^{\overline{m}_A - 1} & \cdots & k_2 & 1 & -f_2 k_2^{\overline{m}_B - 1} & \cdots & -f_2 k_2 & -f_2 \\ \vdots & & & & \vdots & & & \vdots \\ k_{\overline{m}}^{\overline{m}_A - 1} & \cdots & k_{\overline{m}} & 1 & -f_{\overline{m}} k_{\overline{m}}^{\overline{m}_B - 1} & \cdots & -f_{\overline{m}} k_{\overline{m}} & -f_{\overline{m}} \end{pmatrix} \cdot \begin{pmatrix} p_1 \\ \vdots \\ p_{\overline{m}_A} \\ q_1 \\ \vdots \\ q_{\overline{m}_B} \end{pmatrix} = \begin{pmatrix} f_1 k_1^{\overline{m}_B} - k_1^{\overline{m}_A} \\ f_2 k_2^{\overline{m}_B} - k_2^{\overline{m}_A} \\ \vdots \\ f_{\overline{m}} k_{\overline{m}}^{\overline{m}_B} - k_{\overline{m}}^{\overline{m}_A} \end{pmatrix} \tag{5}$$

is a necessary and sufficient[2] condition for a monic rational function of degrees $(\overline{m}_A, \overline{m}_B)$ to satisfy the support set $\{(k_1, f_1), \ldots, (k_{\overline{m}}, f_{\overline{m}})\}$.

Since $P(Z)/Q(Z)$ satisfies the support set $\{(k_1, f_1), \ldots, (k_{\overline{m}}, f_{\overline{m}})\}$, Equation (5) must have a solution. A monic rational function $P'(Z)/Q'(Z)$ with degrees $(\overline{m}_A, \overline{m}_B)$ that satisfies Equation (5) can therefore be found using classical Gaussian elimination in $O(m^3)$ operations.
$P'(Z)/Q'(Z)$ and $P(Z)/Q(Z)$ must satisfy the same support set V, so by Theorem 3 they must be equivalent. Since by construction $P(Z)/Q(Z)$ reduces to $f(Z)$, $P'(Z)/Q'(Z)$ must reduce to $f(Z)$ as well. Thus, if $g(Z)$ is the greatest common divisor of $P'(Z)$ and $Q'(Z)$, then $f(Z)$ is equal to

$$\frac{P'(Z)/g(Z)}{Q'(Z)/g(Z)}$$

Note that the degree of $g(Z)$ is necessarily $\overline{m}_A - m_A$, and so the degrees of the recovered rational function are (m_A, m_B), as is expected.

3.3 Choosing Sample Points

If a sample point k is chosen that is an element of S_A or S_B then the corresponding characteristic polynomial will vanish. Such *anomalous evaluation points* complicate the problem interpolation, especially when k is an element of both S_A and S_B. There are a number of different ways of dealing with this problem. In the following, we will show one of them, which we call the Expanded Finite Field approach.

[2]Because the rational function being recovered is monic, all solutions to Equation (5) represent valid rational functions of the proper degrees. In particular, neither numerator nor denominator can be the polynomial 0. Thus, the satisfying Equation (5) is a sufficient condition as well.

The Expanded Finite Field approach works by ensuring that the chosen sample points are not elements of S_A or S_B, independent of the contents of S_A and S_B. This is done by choosing sample points from a different set of elements than is used for S_A and S_B. Recall that the elements of S_A and S_B are b-bit strings, which are mapped into elements of the finite field $\mathbb{F}_q$. By enlarging the field $\mathbb{F}_q$, we can ensure the existence of a subset of $\mathbb{F}_q$ that is not used to encode the b-bit strings. For instance, we could use $q = 2^{b+1}$ instead of $q = 2^b$. Recall that elements of $\mathbb{F}_{2^{b+1}}$ have the form

$$a = a_0 + a_1\alpha + a_2\alpha^2 + \cdots + a_b\alpha^b,$$

where the a_i are either 0 or 1 and α is a zero of an irreducible polynomial of degree $b+1$ over $\mathbb{F}_2$. Then b-bit strings can be mapped into elements of $\mathbb{F}_{2^{b+1}}$ where $a_b = 0$. The sample points can then be chosen such that $a_b = 1$ without colliding with any elements of S_A or S_B.

For the case where q is an odd prime number greater than 2^b, we can map all possible b-bit strings onto the elements 0 through $2^b - 1$. Thus, the elements 2^b through $q-1$ can be used as sample points without colliding with any elements of S_A or S_B. In this case, q must be chosen large enough to accommodate the extra sample points.

In either case, the storage cost of enlarging $\mathbb{F}_q$ to include sample points disjoint from the elements of S_A and S_B is at most one bit per number. Note that the Expanded Finite Field approach allows the parties to choose their evaluation points *a priori*, without knowledge of the contents of any data sets.

3.4 A Complete Protocol

This section describes a complete set reconciliation protocol (Protocol 3) based on the techniques introduced in Sections 3.2 and 3.3.

Most of the calculations required for set reconciliation, including the interpolation and factoring, only depend on the size of the symmetric difference between the sets to be reconciled. Evaluating each host's characteristic polynomial at a given sample point, however, requires a linear scan over that host's data set. Fortunately, this cost can be amortized over the updates to the data sets. This approach is taken in Protocol 3. Accordingly, the protocol consists of three methods: `addElement`(*elt*) and `removeElement`(*elt*), which are called whenever a host updates its local data set, and `reconcile`(), which computes Δ_A and Δ_B.

Following the Expanded Finite Field approach described in Section 3.3, Protocol 3 assumes that there is a set $E \stackrel{def}{=} \{k_1, k_2, \ldots, k_{\overline{m}}\} \subseteq \mathbb{F}_q$ of evaluation points that is agreed upon *a priori* between A and B. The set E is assumed not to overlap with the representation in $\mathbb{F}_q$ of any length-b bit-string.

Each host maintains a vector of reconciliation data *recData* of length $\overline{m}$, such that for host A *recData*[i] is the value of $\chi_{S_A}(Z)$ evaluated at k_i, and similarly for B. The `addElement`() and `removeElement`() methods incrementally maintain *recData* as elements are added and removed from S_A and S_B. A count *setSize* of the number of elements in the data set is also maintained for each host.

The core of the protocol is the `reconcile`() method. Here, the values of *recData* at A and B are combined to compute the values of $\chi_{S_A}(Z)/\chi_{S_B}(Z)$ at the evaluation points. Then, using the methods of Section 3.2, a monic rational function is found such that the degrees of the numerator sum to no more than $\overline{m}$ and their difference is $A.setSize - B.dataSetSize = |S_A| - |S_B|$. This rational function converted to its reduced form is equal to $\chi_{\Delta_A}(Z)/\chi_{\Delta_A}(Z)$. The numerator and denominator are then factored to recover Δ_A and Δ_B. A simple and efficient factoring algorithm is provided in Appendix A as well as references to other more efficient techniques.

The following example demonstrates Protocol 3 concretely.

Example 2 Consider the sets $S_A = \{1, 2, 9, 12, 33\}$ and $S_B = \{1, 2, 9, 10, 12, 28\}$ stored as 6-bit integers at hosts A and B respectively. We treat the members of S_A and S_B as members of the finite field $\mathbb{F}_{97}$ Assume that $\overline{m} = 5$ is an upper bound on the size of the symmetric difference between S_A and S_B.

Let the set of evaluation points E be $\{-1, -2, -3, -4, -5\}$. Since $97 > 2^6 + \overline{m}$, the elements of E don't coincide with the representation of any length-6 bit-string. Assume that `addElement`(x) is called at A for every $x \in S_A$ and at B for every $x \in S_B$.

The characteristic polynomials for A and B are:

$$\begin{aligned}\chi_{S_A}(Z) &= (Z-1)\cdot(Z-2)\cdot(Z-9)\cdot(Z-12)\cdot(Z-33),\\ \chi_{S_B}(Z) &= (Z-1)\cdot(Z-2)\cdot(Z-9)\cdot(Z-10)\cdot\\ &\qquad (Z-12)\cdot(Z-28).\end{aligned}$$

The following table shows the values at the evaluation points of the characteristic polynomials and the value of their ratio. Recall that all calculations are done over $\mathbb{F}_{97}$.

$Z =$	-1	-2	-3	-4	-5
$\chi_{S_A}(Z)$	58	19	89	77	4
$\chi_{S_B}(Z)$	15	54	68	77	50
$\chi_{S_A}(Z)/\chi_{S_B}(Z)$	75	74	17	1	35

The values of $\chi_{S_A}(Z)$ and $\chi_{S_A}(Z)$ are computed incrementally by the calls to `addElement()`. The values of $\chi_{S_A}(Z)/\chi_{S_B}(Z)$ are computed in the `reconcile()` method.

Using the techniques of Section 3.2, we can instantiate and solve Equation 5 to find a rational function such that the sum of the degrees of the numerator and denominator is 5 and the difference is $d = |S_A| - |S_B| = -1$. Since the actual symmetric difference is less then 5, Equation 5 is singular, *i.e.* there is more than one solution. Fortunately, any solution will do. Consider, for example, the following solution to Equation 5.

$$\frac{Z^2 + 46Z + 12}{Z^3 + 41Z^2 + 91Z + 4}$$

The GCD of the numerator and the denominator is $(Z - 18)$. By dividing out the GCD we get:

$$\frac{Z - 33}{Z^2 + 59Z + 86}$$

The zeros of the numerator and denominator are $\{33\}$ and $\{10, 28\}$ respectively, which are exactly equal to Δ_A and Δ_B.

3.4.1 Analysis

In order to compute the sets Δ_A and Δ_B, Protocol 3 needs access to the values of *setSize* and *recData*[] at both A and B. Assuming that `reconcile()` is executed by B, then only A's data needs to be obtained. Thus, B needs to obtain $\overline{m}$ elements of $\mathbb{F}_q$ plus an extra b bits to specify *setSize*. According to the Expanded Finite Field approach of Section 3.3, q can be chosen so that $q \leq 2^{b+1}$. To complete the reconciliation, B must send to A the contents of Δ_A, which requires the transmission of an extra m_A length-b bit-strings. This leads to a total communications complexity of:

$$(b+1)\overline{m} + b + bm_A = (\overline{m} + m_A + 1)b + \overline{m}$$

If $\overline{m}$ is chosen near $m = m_A + m_B$, then this is just over twice the cost of simply sending the missing vectors. Section 4 compares these results to the information theoretic bounds.

The computational complexity of Protocol 3 has two components: the cost of evaluating the characteristic polynomials $\chi_{S_A}(Z)$ and $\chi_{S_B}(Z)$ at the sample points, and the cost of interpolating and factoring. The cost of evaluating the characteristic polynomial of the set S_A at $\overline{m}$ sample points is $O(|S|\overline{m})$. However, the values of the characteristic polynomials can be maintained incrementally as the set S_A is built up, with a computational cost of $O(\overline{m})$ per insertion and deletion, as was done in Protocol 2. Moreover, the computation of the sample values does not have to be redone for every run of the reconciliation protocol.

The cost of interpolation using Gaussian elimination to solve the system of linear equations is $O(\overline{m}^3 \lg q)$ bit operations. It may be possible to reduce the exponent using fast linear equation solvers, or by choosing special evaluation points (as is done for FFT), but the fact that the linear equations may be singular makes these approaches somewhat difficult.

The cost of root finding using the simple algorithm given in Appendix A is $O(\overline{m}^3 \lg q)$ bit operations. Again asymptotically faster algorithms may be used to improve the exponent, but both linear equation solving and root finding exponents must be reduced to improve the asymptotic complexity of $O(\overline{m}^3 \lg q)$.

3.5 Probabilistic Verification

The discussion until now has assumed that there is a known bound $\overline{m}$ on m. In the absence of such a bound, the hosts need to detect that enough samples have been taken to recover the rational function $\chi_{\Delta_A}(Z)/\chi_{\Delta_B}(Z)$. The following theorem suggests how this might be done.

Theorem 4 *Let $f(Z)$ and $g(Z)$ be distinct monic rational functions over $\mathbb{F}_q$ such that the sum of the degrees of the numerator and the denominator is no more than m. If k is a randomly chosen element of some subset $E \subseteq \mathbb{F}_q$, then the probability that $f(Z)$ and $g(Z)$ take on the same values at $Z = k$ is less than or equal to $(m-1)/|E|$.*

This follows from the fact that distinct monic rational functions with the given degree bounds cannot agree on more than $m-1$ points, which follows from Theorem 3.

Theorem 3 can be used to test whether the rational function recovered from a given set of sample points is in fact equal to $\chi_{\Delta_A}(Z)/\chi_{\Delta_B}(Z)$. In particular, let $f(Z)$ be $\chi_{\Delta_A}(Z)/\chi_{\Delta_B}(Z)$ and $g(Z)$ be a rational function reconstructed from some number of samples of $f(Z)$. Furthermore, assume that we're using the Extended Finite Field approach with $q = 2^{b+1}$. Choose E to be the size 2^b subset of $\mathbb{F}_q$ not used for representing the length-b bit-strings. By Theorem 4, the reconciling hosts can verify whether $f(Z) = g(Z)$ by testing whether the two functions are equal on randomly selected elements of E. Note that $|S_A|$ and $|S_B|$ are trivial upper bounds on m_A and m_B respectively, and so the probability of $f(Z)$ and $g(Z)$ being equal on a randomly selected point when the two functions are not equal is bounded above by $\rho = (|S_A| + |S_B| - 1)/2^b$, which is typically very small.

The approach described above requires sample points to be chosen at random, so both the value and the sample point need to be sent for each sample, roughly doubling the number of bits transmitted. In practice, a pseudo-random number generator could be used, allowing for a seed to be sent instead of sending each individual sample point.

The question remains of what protocol to use for transmitting sample points. One approach would be for A to send sample points one at a time. Thus, B would maintain a rational function $g(Z)$ matching the sample points received so far. When B receives k samples in a row that confirm the previous value of $g(Z)$, then B accepts $g(Z)$ as equal to $f(Z)$, and factors the numerator and denominator to recover Δ_A and Δ_B. Until this happens, B continues to request more sample points from A.

The probability that the above protocol terminates with $g(Z) \neq f(Z)$ is bounded above by $m\rho^k$, where k is the number of samples taken. This is because the probability of getting a sequence of k matching sampled values in a row starting at the i-th sample when the two functions are not equal is ρ^k. Since there are no more than m possible starting points for such a sequence, the overall probability is less than $m\rho^k$. This means that, for any given probability ϵ, if k is chosen to be $\lceil \log_\rho(\epsilon/m) \rceil$, the probability of failure is less than ϵ. Since m is not known *a priori*, we can use the weak upper bound of $|S_A| + |S_B|$ to give a corresponding value of

$$k = \lceil \log_\rho(\epsilon/(|S_A| + |S_B|)) \rceil. \tag{6}$$

Thus, for example, to achieve a confidence level of 10^{-12} when reconciling sets of 32-bit bit-strings whose combined size is less than 10000 would require 3 confirming samples.

The number of bits transmitted under this approach is bounded above by:

$$2(b+1)m + b + bm_A + m + k$$

The extra $m + k$ bits is due to the 1-bit messages that B needs to tell A to send the next sample. The computational complexity is thus $O(m^4 \lg q)$, since the root finding needs to be repeated every round.

The approach of sending a single sampled value at a time has the advantage of sending the minimum number of samples. It has the disadvantage, however, of requiring $2(m+k)$ messages. We can reduce the number of messages to $2\lceil \log_c(m+k) \rceil$ by increasing the number of samples sent each round by a factor of c. Then, in the worst case scenario, the number of extra samples sent is $c(m+k)$. Note that Equation (6) can be used for selecting k in this case as well.

The communications complexity for this approach is bounded above by

$$2(b+1)c\overline{m} + b + bm_A + \lceil \log_c(m+k) \rceil$$

which is approximately c times the communications complexity of sending one sample at a time. The computational complexity is $O(\lceil \log_c(m+k) \rceil m^3 \lg q)$.

4 Information-Theoretic Bounds

The set reconciliation algorithms described in Sections 2 and 3 all have communications complexity on the order of bm. In the following, we will show that bm is close to the best achievable communications complexity for both the subset reconciliation problem and for the general set reconciliation problem.

Let $N = |S_A \cap S_B|$. Solving set reconciliation demands that host A discern m_B integers from the $2^b - N - m_A$ that it might be missing. Symmetrically, host B must discern m_A bit-strings among $2^b - N - m_B$ possibilities. This gives the following information-theoretic lower bound on I_{trans}, the number of bits that need to be transmitted between A and B for reconciliation:

$$I_{\text{trans}} \geq \lg\left[\binom{2^b - N - m_A}{m_B} \cdot \binom{2^b - N - m_B}{m_A}\right]. \tag{7}$$

If $m = m_A + m_B$ is held constant then this expression is minimized when m_A or m_B is zero, as appropriate. This follows from the following well-known identity (s.f.r. [18]):

$$\binom{n}{j}\binom{n-j}{k} \geq \binom{n}{j+k}, \tag{8}$$

which is true whenever $n, j, k \geq 0$ and $j + k \leq n$. Assuming, without loss of generality, that $|S_A| \geq |S_B|$ then substituting $n = 2^b - N - m_A$, $j = m_B$, and $k = m_A$ eventually yields:

$$I_{\text{trans}} \quad \geq \quad \lg\left[\binom{2^b - N - m}{m}\right]. \tag{9}$$

When 2^b is at least twice as large as either host set, then the lower bound in inequality (9) becomes $(b - 1 - \lg m) \cdot m \approx bm - m \lg m$. Thus,

$$\frac{I_{\text{trans}}}{bm} \geq 1 - \frac{\lg m}{b}.$$

Typically, $\lg m$ is significantly smaller than b, and so I_{trans} is at best within a small fraction of bm.

5 Related work

Set reconciliation is closely to the problem of error correction over a noisy channel. The main difference between set reconciliation and error correction is that, in the most common model for error correction, every data element has an index corresponding to its place in the transmission order, and errors consist of in-place replacement of one datum with another. This index is stable, in the sense that transmission errors on some elements do not affect the indices of the remaining elements. In set reconciliation, there is no such stable index.

Error correction codes for in-place errors can be applied to the problem of set-reconciliation. In particular, if we choose some ordering on the set $\mathcal{B}_b$ of all length-b bit-strings, then a set $S \subseteq \mathcal{B}_b$ can be represented as a single length-2^b bit-string, where the ith bit of the bit-string is 1 iff the ith element of $\mathcal{B}_b$ is in S. Thus, differences between two sets result in bitwise errors in the corresponding bit-string representations. The length of this representation, however, makes this approach computationally infeasible.

The spurious error correction model, introduced by Levenshtein [19], allows for errors to be insertions or deletions of letters in addition to in-place replacements. A spurious error correction algorithm can be applied to set reconciliation by treating a set S as a string consisting of the elements of S listed in lexicographic order. Insertions and deletions from S then correspond to insertions and deletions from the sorted string.

Many methods have been proposed in the literature for both traditional error correction and spurious error correction [20, 21, 22, 23, 24, 25, 26, 27, 28]. In line with the former model, Metzner and Kapturowski [24] examined the problem of correcting disagreeing pages between two different versions of a file using a minimum communication complexity. They presented an algorithm that, with high probability, corrected most cases of up to Δ disagreeing pages using a single message of Δ signatures, where the size of each signature was logarithmic in the overall file size n. Abdel-Ghaffar and Abbadi [26] used Reed-Solomon codes [29] to provide a deterministic algorithm for the same problem that required at most 2Δ signatures.

In the direction of spurious error correction, Schwarz, Bowdidge, and Burkhard [25] extended Metzner and Kapturowski's result to the case of extraneous or missing pages. Using a divide-and-conquer algorithm, they developed a multi-round protocol for reconciling corrupted copies of a file. Recently, Cormode, Paterson, Sahinhalp, and Vishkin [27] provided a probabilistic algorithm that asymptotically requires $O(m \lg(n/m) \lg \overline{m})$ communication bits for a bound $\overline{m}$ on the size m of the difference between two length-n files.

In this paper, we solved the set reconciliation problem using a single message of roughly $\overline{m}(2b+1)$ bits to reconcile two (arbitrarily large) sets that differ by at most $\overline{m}$ length-b bit-strings. We also provided a comparable probabilistic protocol when a bound $\overline{m}$ is not known. It is important to note, however, that our results are not strictly comparable to these related results.

6 Future Work

Set reconciliation can be applied to a number of different reconciliation problems. One important example is the reconciliation of sets where the data elements are variable length bit-strings. This case can be dealt with by running the set reconciliation algorithm on hashes of the actual data elements instead of the data elements themselves. The result of the reconciliation can then be used to determine which data elements need to be transferred. The problem with the hash approach is that it cannot be done non-interactively, since it has a second stage after the reconciliation where the actual missing data is sent. Another alternative is to break variable length data elements into fixed-length chunks, and to annotate those chunks with indexing information that allows the original elements to be reconstructed from the chunks. These ideas need to be investigated in more detail.

Another common class of reconciliation problem is the reconciliation of (key,value) databases. Such a database can be reconciled by treating each (key,value) pair as a single (possibly variable-length) bit-string. However, it may be possible to take advantage of the fact that in most cases, the keys of such a database change more frequently than the values. It may be useful to consider hybrid protocols that combine set reconciliation with existing error-correction techniques for fixing in-place corruptions of ordered data.

The probabilistic scheme in Section 3.5 also needs further work. The analysis of the convergence of that scheme is conservative and can be substantially improved. Also, we believe that it is possible to augment the probabilistic algorithm we presented so as to converge much more quickly.

Protocol 3 has been implemented using Victor Shoup's well known NTL package[30]. Work is in progress to investigate various applications and to see how effective these protocols can be in practice in reducing the communication load of gossip protocols.

The key to both protocols is the representation of a set by its characteristic polynomial. This use of polynomials is reminiscent of the use of polynomials in Shamir's secret sharing protocol [31]. The relationship between secret sharing and set reconciliation also deserves further study.

7 Conclusion

We have examined the problem of reconciling two related sets, stored at separate hosts, with low communication complexity. We have presented a protocol for the set reconciliation generalized from a special case in which the data stored at one host is a subset of the data stored at the other host.

Perhaps the most surprising result in the paper is the fact that these protocols can be used non-interactively if given a bound on the number of elements that differ between the two hosts. Moreover, the communication complexity of these protocols is remarkably close to the complexity of set reconciliation when each host knows *a priori* which elements the other host is missing.

8 Acknowledgments

We are grateful to Ramin Takloo-Bighash, Edward Reingold, Fred Schneider, Lazar Trachtenberg, and Alexander Vardy for stimulating discussions. The implementation was developed with the help of Eyal Adler, and we would like to thank Graham Cormode for pointing us to various related works.

References

[1] Alan J. Demers, Daniel H. Greene, Carl Hause, Wes Irish, and John Larson, "Epidemic algorithms for replicated database maintenance," in *Proceedings of the Sixth Annual ACM Symposium on Principles of Distributed Computing*, Vancouver, British Columbia, Canada, August 1987, ACM, number 6, pp. 1–12.

[2] R. Adams, "RFC1036: Standard for interchange of USENET messages," December 1987.

[3] Richard Andrew Golding, *Weak-Consistency Group Communication and Membership*, Ph.D. thesis, UC Santa Cruz, December 1992, Published as technical report UCSC-CRL-92-52.

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

[5] P. Mockapetris, "RFC1034: Domain names - concepts and facilities," November 1987.

[6] Robbert van Renesse, Yaron Minsky, and Mark 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.

[7] Robbert van Renesse, "Captain cook: A scalable navigation service," In preparation.

[8] Katherine Guo, Mark Hayden, Robbert van Renesse, Werner Vogels, and Kennth P. Birman, "Gsgc: An efficient gossip-style garbage collection scheme for scalable reliable multicast," Tech. Rep., Cornell University, December 1997.

[9] Mark Hayden and Kenneth Birman, "Probabilistic broadcast," Tech. Rep., Cornell University, 1996.

[10] G.H. Hardy and E.M. Wright, *An Introduction to the Theory of Numbers*, Oxford University Press, 1954.

[11] A. Trachtenberg and Y. Minsky, "Efficient reconciliation of unordered databases," Tech. Rep., Cornell University, 1999.

[12] P. Naudin and C. Quittê, *Theoretical Computer Science*, vol. 191, chapter Univariate Polynomial Factorization over Finite Fields, pp. 1–36, Elsevier Science B.V., 1996.

[13] V. Shoup, "Factoring polynomials over finite fields: Asymptotic complexity vs. reality," in *Proc. IMACS Symposium*, Lille, France, 1993.

[14] V. Shoup, "A new polynomial factorization algorithm and its implementation," *Journal of Symbolic Computation*, vol. 20, pp. 363–397, 1995.

[15] Richard Eliot Zippel, *Effective Polynomial Computation*, Kluwer Academic Press, Boston, 1993.

[16] E. Kaltofen and V. Shoup, "Subquadratic-time factoring of polynomials over finite fields," in *27th Annual ACM Symposium on Theory of Computing*, 1995, vol. 9, pp. 398–406.

[17] Josef Stoer and Roland Bulirsch, *Introduction to Numerical Analysis*, Springer-Verlag, New York, 2nd edition, 1993.

[18] T.H. Cormen, C.E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, MIT Press, 1990.

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

[20] D. Barbará, H. Garcia-Molina, and B. Feijoo, "Exploiting symmetries for low-cost comparison of file copies," *Proceedings of the International Conference on Distributed Computing Systems*, pp. 471–479, 1988.

[21] D. Barbará and R.J. Lipton, "A class of randomized strategies for low-cost comparison of file copies," *IEEE Transactions on Parallel Distributed Systems*, pp. 160–170, April 1991.

[22] W. Fuchs, K.L. Wu, and J.A. Abraham, "Low-cost comparison and diagnosis of large remotely located files," *Proceedings of the Symposium on Reliability in Distributed Software and Database Systems*, pp. 67–73, 1996.

[23] J.J. Metzner, "A parity structure for large remotely located replicated data files," *IEEE Transactions on Computers*, vol. C-32, no. 8, pp. 727–730, August 1983.

[24] J.J. Metzner and E.J. Kapturowski, "A general decoding technique applicable to replicated file disagreement location and concatenated code decoding," *IEEE Transactions on Information Theory*, vol. 36, pp. 911–917, July 1990.

[25] 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.

[26] 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.

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

[28] A.V. Evfimievski, "A probabilistic algorithm for updating files over a communication link," *Theoretical Computer Science*, pp. 191–199, 2000.

[29] F.J. MacWilliams and N.J.A. Sloane, *The Theory of Error-Correcting Codes*, North-Holland Publishing Company, New York, 1977.

[30] V. Shoup, "Ntl: A library for doing number theory," http://www.shoup.net.ntl/.

[31] Adi Shamir, "How to share a secret," *CACM*, vol. 22, pp. 612–613, Nov. 1979.

A Root Finding of Polynomials

Assume we are given a polynomial $f(Z)$ of degree d over a finite field $\mathbb{F}_q$. This appendix shows how to determine if all the zeros of $f(Z)$ are distinct and lie in $\mathbb{F}_q$ and, if so, how to find them quickly. We show how to use classical algorithms to perform zero finding in expected $O(d^3 \lg q)$ field operations. More sophisticated algorithms improve the asymptotic complexity to as low as $O(d^{1.82} \lg q)$ [16], although their basic structure is similar to that presented here and their practical benefits are not clear. All of these results are well known (see for instance, [12, 13, 14, 15]). They are included here for completeness.

The particular type of root finding needed by the set reconciliation protocols involves three steps. First, determine if $f(Z)$ is square free. Second, verify that all irreducible factors of $f(Z)$ are linear. And finally, find the linear factors of $f(Z)$.

We can determine if $f(Z)$ is square-free by computing the GCD (greatest common divisor) of $f(Z)$ and its derivative $f'(Z)$. Using the Euclidean algorithm and classical polynomial algorithms, this can be done in $O(d^2)$ operations in $\mathbb{F}_q$. Verifying that $f(Z)$ is the product of d linear factors can also be done by computing GCD's. Note that all elements of $\mathbb{F}_q$ are zeros of $Z^q - Z$. Thus, $Z^q - Z$ is the product of monic linear polynomials over $\mathbb{F}_q$. If $f(Z)$ is square free, it is the product of a linear polynomials only if $f(Z)$ divides $Z^q - Z$. Checking for such a divisibility using classical division would require $O(d \cdot q)$ operations. However, this complexity can be reduced to $O(d^2 \lg q)$ by using repeated squaring to compute $h(Z) \equiv Z^q \pmod{f(Z)}$ and then testing if $h(Z) - Z \equiv 0 \pmod{f(Z)}$. The repeated squaring stage dominates the time complexity, since $O(\lg p)$ squarings are required, and each squaring involves a polynomial multiplication followed by finding the remainder mod $f(Z)$. Using classical algorithms each multiply and remainder takes $O(d^2)$ time. Thus, $O(d^2 \lg p)$ field operations will be required in all for the verification of divisibility.

Finally, we need to find the linear factors of $f(Z)$. This is done using probabilistic techniques as follows. We consider two different cases for the field $\mathbb{F}_q$ (corresponding to the possible choices for use in our set-reconciliation protocols): one where q is a prime and the other where $q = 2^\ell$. When q is a prime, note that the elements of $\mathbb{F}_q$ are zeros of

$$Z^q - Z = (Z^{\frac{q-1}{2}} + 1) \cdot Z \cdot (Z^{\frac{q-1}{2}} - 1).$$

So, almost half of the elements of $\mathbb{F}_q$ are zeros of $R(Z) = Z^{\frac{q-1}{2}} - 1$.

A polynomial with similar properties can also be constructed for the field $\mathbb{F}_{2^\ell}$. Denote by $R(Z)$ the polynomial

$$R(Z) = Z^{2^{\ell-1}} + Z^{2^{\ell-2}} + \cdots + Z^4 + Z^2 + Z.$$

Over the field $\mathbb{F}_{2^\ell}$, we have

$$\begin{aligned} R(Z)\cdot(R(Z)+1) &= R(Z)^2 + R(Z), \\ &= Z^{2^\ell} + Z^{2^{\ell-1}} + \cdots + Z^2 + R(Z), \\ &= Z^{2^\ell} + Z. \end{aligned}$$

So, all the elements of $\mathbb{F}_{2^\ell}$ are zeros of $R(Z)\cdot(R(Z)+1)$, and each element is either a zero of $R(Z)$ or of $R(Z)+1$.

To determine the zeros of $f(Z)$, we chose a random element of $a \in \mathbb{F}_q$ and compute the greatest common divisor of $f(Z)$ and $R(Z-a)$, which will have almost half the degree of $f(Z)$. Applying this technique recursively on the two factors of $f(Z)$, with different values for a will further split the polynomial, ultimately into linear factors. In total, the expected number of GCD required will be $O(d)$. For odd q, the first GCD is done via repeated squaring in modulus, as in the previous paragraph. For $q = 2^\ell$, the remainder of each of the terms of $R(Z)$ is computed (from lowest to highest degree) and then summed.

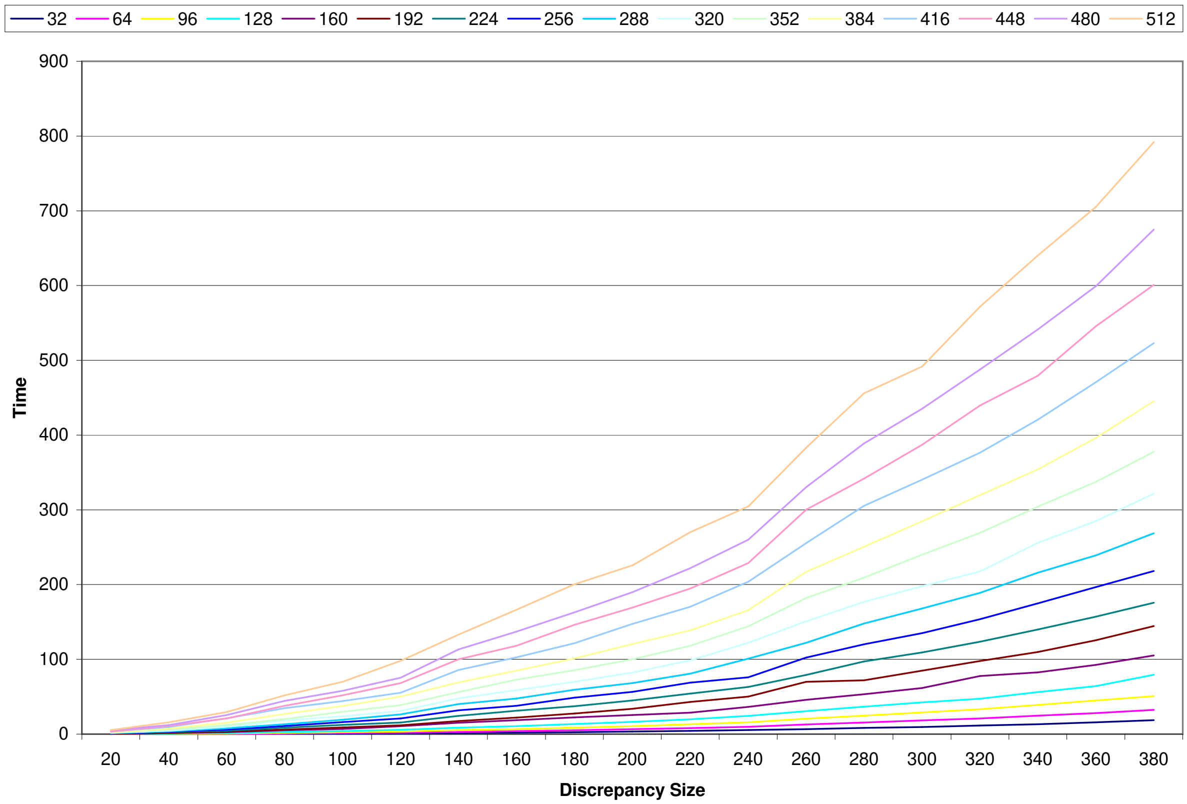
B Implementation of Protocol 3

The first question to be answered in any implementation of the set reconciliation protocol is what finite field to use. Since the data sets, consist of bit strings of length b, the most efficient field to use, information theoretically, would be one of the form $\mathbb{F}_{2^m}$. Unfortunately, arithmetic in these fields can be quite slow, when special hardware support is not available. Thus, it is preferable to use $\mathbb{F}_q$, where $q > 2^b$ is a prime number.

We have implemented the set reconciliation protocol 3 using Shoup's well-known NTL package [30]. Instead of using the Expanded Finite Field approach, the implementation instead does some extra bookkeeping to handle anomalous evaluation points. The bookkeeping information is 1 bit per sample point, so it has the same communication complexity as the protocol described in the paper. The implementation also uses a finite field with an odd number of elements. Table 1 shows the number of seconds of computation time required by this protocol on a 550Mhz Pentium III processor for different size data set words, and different numbers of discrepancies. In this testing, we have assumed that the reconciliation data was maintained incrementally and thus did not measure the time required to compute the values of the characteristic polynomials.

Since the word sized used for the data set elements is a multiple of 32, the the prime numbers used in the finite field arithmetic were always one word larger than the data words. This is typical of most applications of the protocol, since users would most likely not appreciate having their data set elements restricted to 31, 63 or 127 bits.

It is worth noting that even with fairly large data set words (256 bits), and large numbers of discrepancies (200), the amount of time required to compute the missing elements is relatively small (less than a minute) and that the growth in time complexity is less than cubic in the size of the discrepancy.



b / Δ	40	80	120	160	200	240	280	320	360
32	0.078	0.391	0.907	1.984	3.438	5.406	8.219	11.610	15.875
64	0.219	0.921	1.922	4.093	6.532	9.844	15.360	21.031	28.187
96	0.453	1.719	3.406	6.734	10.516	15.704	24.672	33.391	44.953
128	0.750	2.953	5.672	10.579	16.406	24.375	36.656	47.375	64.250
160	1.359	5.110	10.547	18.703	25.375	36.422	53.422	77.828	92.750
192	1.453	6.078	11.688	22.031	33.703	50.266	72.031	97.750	125.656
224	2.734	8.531	15.578	31.218	45.015	63.110	97.328	123.672	157.062
256	3.046	10.985	20.875	37.875	56.406	75.985	120.141	153.610	196.797
288	3.703	13.265	26.453	47.656	68.234	100.906	148.047	188.906	239.125
320	4.547	18.453	30.969	58.891	82.390	122.203	176.969	217.671	285.000
352	5.969	20.172	38.641	72.875	100.109	144.187	209.421	269.156	337.406
384	7.469	26.828	50.063	84.938	120.312	165.828	250.672	319.782	396.250
416	9.563	34.766	55.469	102.609	147.422	203.797	305.36	376.579	470.765
448	10.765	37.813	68.219	118.109	169.281	228.922	341.907	439.407	545.718
480	12.078	44.421	75.328	137.125	189.750	259.984	388.75	487.688	599.297
512	15.844	51.969	97.985	166.297	225.656	304.421	455.813	571.750	705.359

Figure 1: Computational time needed to reconcile Δ vectors each of size b bits on a 550MHz Pentium III processor.

Protocol 3 Set reconciliation protocol given upper bound on m

Each host evaluates the following functions, assuming that $\overline{m}$ is an upper bound on m, the size of the symmetric difference between S_A and S_B. We also assume that hosts A and B agree on a field $\mathbb{F}_q$, and that $E = \{k_1, k_2, \ldots, k_{\overline{m}}\} \subset \mathbb{F}_q$ is some set of size $\overline{m}$ that does not overlap with S_A or S_B. The notation $H.var$ is used to refer to the value of var at host H.

We initialize *setSize* to 0 and $recData[i]$ to 1, for all i from 1 to $\overline{m}$.

`addElement`(*elt*)

1. Increment *setSize* by 1.
2. For each $i = 1 \ldots \overline{m}$ compute

$$recData[i] = recData[i] \times (k_i - elt)$$

over the field $\mathbb{F}_q$.

`removeElement`(*elt*)

1. Decrement *setSize* by 1.
2. For each $i = 1..\overline{m}$ compute

$$recData[i] = recData[i]/(k_i - elt)$$

over the field $\mathbb{F}_q$.

`reconcile`()

1. Initialize $d = A.setSize - B.setSize$.
2. Define the support set V to be:

$$\{(k_i, A.recData[i]/B.recData[i]) \mid k_i \in E\}.$$

Then, using the technique described in Section 3.2, find a monic rational function $P(Z)/Q(Z)$ that satisfies V such that:

$$degree(P(Z)) + degree(Q(Z)) \leq \overline{m}$$
$$degree(P(Z)) - degree(Q(Z)) = d$$

3. Compute Δ_A and Δ_B as the zeros of the normalized polynomials $\frac{P(Z)}{\mathrm{GCD}(P(Z),Q(Z))}$ and $\frac{Q(Z)}{\mathrm{GCD}(P(Z),Q(Z))}$ respectively.