# Set Reconciliation with Nearly Optimal Communication Complexity[*]

**Yaron Minsky**
Cornell University
*yminsky@cs.cornell.edu*

**Ari Trachtenberg**
University of Illinois
*arit@alum.mit.edu*

**Richard Zippel**
Interdisciplinary Center Herzliya
*rz@idc.ac.il*

April 25, 2000

**Abstract**

We consider a fundamental problem that arises in the context of gossip protocols. Specifically, we consider the problem of efficiently reconciling two similar sets held by different hosts while minimizing the communication complexity. We provide two surprisingly simple and efficient protocols that exhibit tractable computational complexity and nearly optimal communication complexity. These protocols can be adapted to work over a broadcast channel, allowing many clients to reconcile with one host based on a broadcasted signal. Thus, an arbitrary number of clients each of whose data differs from that of the host by no more than $N$ bits can be reconciled by a single broadcast of $O(N)$ bits, independent of the the number of clients and independent of the size of the data sets.

[*]A version of this paper is available as Technical Report 2000-1796 from Cornell University

## 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 [3, 1], 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 [5, 7, 11, 21]; locations of resources [20]; bibliographic data [5]; and broadcast messages [1, 5, 6, 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-$b$ bitstrings, 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 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 simple and efficient set reconciliation protocols. The communication complexity of these protocols is independent of the sizes of the hosts' sets; it depends only on the size of the difference between the two sets. Moreover, under certain circumstances, set reconciliation can be achieved without interaction, using just a single message. 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 set reconciliation problem is related to the well-known problem of error correction over a noisy channel. The main difference between set reconciliation and traditional error correction is that in the traditional model for error correction each data item has an implicit index corresponding to its place in the transmission order. This index is stable, in the sense that transmission errors on some elements do not affect the indices of the remaining elements. In the context of set reconciliation there is no such stable indexing.

Set reconciliation is more closely related to the $\alpha$-edits problem, which is the problem of reconciling two strings where one string differs from the other by a bounded number of edits, *i.e.* insertions or deletions. The $\alpha$-edits problem was analyzed by Orlitsky [13], and Evfimievski [4] presented a probabilistic solution. An algorithm for solving the $\alpha$-edits problem can be used to perform set reconciliation by treating the hosts' sets as strings consisting of the set elements listed in lexicographic order. Missing elements then correspond to deletions and extra elements to insertions. However, the protocols that result from applying results such as those in [4] to set reconciliation yield solutions that are significantly less efficient than those we present. Note that a solution to the set-reconciliation problem is not necessarily a solution to the $\alpha$-edits problem.

Section 2 sets up our definitions and notation, and also presents information-theoretic bounds on set reconciliation. Section 3 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. Finally, Section 4 generalizes this protocol to the general set reconciliation problem. Both protocols achieve nearly optimal communications complexity and tractable computational complexity.

## 2 Definitions and Bounds

Consider a pair of hosts $A$ and $B$ that each have a set of length-$b$ bitstrings, denoted $S_A$ and $S_B$ respectively. Formally, the problem of set reconciliation is to find a sequence of message exchanges between hosts $A$ and $B$ such that at the end of the exchanges both $A$ and $B$ know the set $S_A \cup S_B$. The problem of *subset reconciliation* is a special case of set reconciliation where it is assumed that $S_B \subseteq S_A$. We will measure the complexity of a solution to these problems according to three criteria: the number of messages sent, the number of bits transmitted, and the computation done by the two hosts.

Naively, set reconciliation can be accomplished by each host sending its entire set to the other host. This simple algorithm would require communication of $(|S_A| + |S_B|)b$ bits. If we denote the differences between the two hosts by $\Delta_A = S_A \setminus S_B$ and $\Delta_B = S_B \setminus S_A$, then this can be rewritten as a communication complexity of

$$2|S_A \cap S_B| + |\Delta_A| + |\Delta_B| \text{ bits.}$$

Our goal in this paper is to describe protocols that accomplish set reconciliation with an exchange of only $O(|\Delta_A| + |\Delta_B|)$ bits, independently of the size of the hosts' original sets.

To simplify our notation, we define $N = |S_A \cap S_B|$, $m_A = |\Delta_A|$, $m_B = |\Delta_B|$ and $m = m_A + m_B$. 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 $m_A$ bitstrings among $2^b - N - m_B$ possibilities. This gives the following information-theoretic lower bound on $I_{\text{trans}}$, the number of bits that need to be transmitted between $A$ and $B$ for reconciliation:

$$I_{\text{trans}} \geq \lg\left[\binom{2^b - N - m_A}{m_B}\right] + \lg\left[\binom{2^b - N - m_B}{m_A}\right]. \tag{1}$$

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. [2]):

$$\binom{n}{j}\binom{n-j}{k} \geq \binom{n}{j+k}, \tag{2}$$

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:

$$I_{\text{trans}} \quad \geq \quad \lg\left[\binom{2^b - N - m}{m}\right]. \tag{3}$$

When $2^b$ is at least twice as large as either host set, then the lower bound Equation (3) becomes $(b - 1 - \log m) \cdot m \approx bm - m\log m$. Surprisingly, we can get very close to this information-theoretic bound for both the set and subset reconciliation problems, as will be shown in the next two sections.

## 3 Subset Reconciliation

One case of the subset reconciliation problem has a straightforward solution—when there is only one bitstring of host $A$ not known to host $B$ (*i.e.* when $m = m_A = 1$). 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 $\Delta_A$ 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 bitstring.

**Protocol 1** Subset Reconciliation when $m_A = 1$

1. Host $A$ computes $\mathsf{parity}_A$, the parity sum of its bitstrings, and sends it to $B$
2. Host $B$ computes $\mathsf{parity}_B$, the parity sum of its bitstrings.
3. Host $B$ computes the parity sum of $\mathsf{parity}_A$ and $\mathsf{parity}_B$, which is precisely the missing bitstring.

## 3.1 Characteristic Polynomials

Protocol 1 is limited to the case where $m_A = 1$ 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 bitstring. 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.

$$\begin{aligned} \chi_S(Z) &= (Z - x_1)(Z - x_2)(Z - x_3)\cdots(Z - x_n) \\ &= Z^n - \sigma_1(S)Z^{n-1} + \sigma_2(S)Z^{n-2} + \cdots + (-1)^n\sigma_n(S). \end{aligned} \tag{4}$$

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

$$\begin{aligned} \sigma_1(S) &= x_1 + x_2 + + \cdots + x_m, \\ \sigma_2(S) &= x_1x_2 + x_1x_3 + \cdots + x_{m-1}x_m, \\ \sigma_3(S) &= x_1x_2x_3 + x_1x_2x_4 + \cdots + x_{m-2}x_{m-1}x_m, \\ &\vdots \\ \sigma_m(S) &= x_1x_2\cdots x_m. \end{aligned}$$

Note that the zeros of $\chi_S(Z)$ are precisely the elements of $S$. Thus, the elements of $S$ can be recovered by factoring $\chi_S(Z)$.

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 3.1 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 $\chi_{S_B}(Z)$ to $B$, the coefficients of $\chi_{\Delta_A}(Z)$ can be reconstructed by $B$.

**Theorem 3.1** *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 $\chi_{\Delta_A}(Z)$:*

$$\sigma_k(\Delta_A) = \sigma_k(S_A) - \sigma_k(S_B) + \left[\sum_{0<i<k} \sigma_i(\Delta_A)\sigma_{k-i}(S_B)\right] \tag{5}$$

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

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

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

$$\begin{aligned}
\sigma_1(S_A) &= \sigma_1(\Delta_A) + \sigma_1(S_B) && (6)\\
\sigma_2(S_A) &= \sigma_2(\Delta_A) + \sigma_1(\Delta_A)\sigma_1(S_B) + \sigma_2(S_B) && (7)\\
&\vdots \\
\sigma_k(S_A) &= \sigma_k(\Delta_A) + \sum_{0<i<k} \sigma_i(\Delta_A)\sigma_{k-i}(S_B) + \sigma_k(S_A). && (8)
\end{aligned}$$

Equation (5) follows from a simple rearrangement of Equation (8) and can be applied iteratively to build up the first $k$ elementary symmetric polynomials of $\Delta_A$ 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-$b$ bitstrings onto numbers. 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 bitstring is then interpreted as a binary integer less than $q$. By Bertrand's Postulate [8, 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 $\alpha$, where $\alpha$ is the zero of an irreducible polynomial of degree $b$ over $\mathbb{F}_2$. In this case, each bitstring is represented using exactly $\lg q = b$ bits.

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-$b$ bitstring. 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 bitstrings as elements of $\mathbb{F}_q$, without specifying a choice of $q$.

## 3.2 The Protocol

Protocol 2, which was first described in [19], 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 highest-degree coefficient of any characteristic polynomial is always 1. Note also that Protocol 2 assumes that $m_A$, the number of missing bitstrings, 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.

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 on hosts A and B respectively. We use the prime number $q = 11$ as the communication modulus, and all calculations are carried out modulo 11 (*i.e.* over $\mathbb{F}_{11}$).*

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

**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 3.1, 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 $\Delta_A$ are precisely the zeros of $\chi_{\Delta_A}(Z)$.

---

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

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

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

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

- *Host $B$ uses Equation* (5) *to compute the elementary symmetric polynomial values of $\Delta_A = S_A \setminus S_B$:*

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

- *Using the equality in Equation* (4)*, 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 $\Delta_A = \{1, 3, 5\}$.*

## 3.3 Analysis

Protocol 2 requires two messages: one to determine the size $m_A$, and the other to transmit $m_A$ required coefficients. If an upper bound on $m_A$ is known, then the protocol requires only a single message, whose length depends to 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 to the client. 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, which is close to the information-theoretic lower bound in Equation (3).

**Theorem 3.2** *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

$$\sigma_i(S_A \cup \{x\}) = \sigma_i(S_A) + x \cdot \sigma_{i-1}(S_A). \tag{9}$$

Thus, in order 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, 16, 17, 22]. Appendix A describes a practical method for factoring a square-free polynomial of degree $m$ in expected time $O(m^3 \log q)$; more sophisticated algorithms [10] can bring this asymptotic time down to $O(m^{1.82} \log q)$, but their practical benefits are not clear.

## 4 Set Reconciliation

Recall that the subset reconciliation algorithm described in Section 3 works by first recovering $\chi_{\Delta_A}(Z)$, the characteristic polynomial of the set of missing bitstrings, and then determining the roots of that polynomial in order to recover the elements of the difference set $\Delta_A$. 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 3 does not apply directly to set reconciliation for two reasons: first, the technique for recovering 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.

### 4.1 Overview

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

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

This holds because terms common to both $A$ and $B$ cancel out in the division. Thus, the degrees of the numerator and denominator of the (reduced) rational function are $m_A$ and $m_B$ respectively.

The broad outline of our approach is as follows. Hosts $A$ and $B$ evaluate $\chi_{S_A}(Z)$ and $\chi_{S_B}(Z)$ respectively at the same $\overline{m}+1$ sample points, where $\overline{m}$ is an upper bound on $m$. The two hosts combine the sampled values so as to compute $\chi_{S_A}(Z)/\chi_{S_B}(Z)$, and therefore $\chi_{\Delta_A}(Z)/\chi_{\Delta_B}(Z)$, at the sample points. The hosts then interpolate the sampled values to recover the coefficients of $\chi_{\Delta_A}(Z)/\chi_{\Delta_B}(Z)$. Finally, by factoring the numerator and the denominator the elements of $\Delta_A$ and $\Delta_B$ are recovered.

Sections 4.2 and 4.3 discuss rational function interpolation and the selection of sample points in more detail. A concrete protocol is then described in Section 4.4. Finally Section 4.5 discusses how set reconciliation can be solved without an *a priori* bound $\overline{m}$ on $m$.

## 4.2 Rational Function Interpolation

A *support set* is a set of pairs $(k_i, r_i)$ where the $k_i$ are distinct. A support set $V$ is *satisfied by a function* $f$ if $f(k_i) = r_i$ for every $(k_i, r_i)$ in $V$. Rational function interpolation is the problem of finding a unique rational function that satisfies a given support set given bounds on the degree of the numerator and the denominator.

In our case, we are interested in recovering only monic rational functions, *i.e.*, rational functions such that the coefficient of the highest-degree term of both the numerator and denominator is 1. The following theorem describes conditions under which a monic rational function can be found that satisfies a given support set, and the complexity of finding that function. The proof is found in Appendix B.

**Theorem 4.1** *Choose $V$ to be a support set of cardinality $m_1 + m_2$. Then, there is a unique monic rational function $f$ that satisfies $V$ whose numerator has degree less than or equal to $m_1$ and whose denominator has degree less than or equal to $m_2$.*

The function $f$ can be determined in $O((m_1+m_2)^3 \lg q)$ time using classical Gaussian elimination, as illustrated in the appendix. Note that Theorem 4.1 requires bounds for the degrees of the numerator and denominator of the rational function. In our context, this translates to bounds on $m_A$ and $m_B$, the sizes of the two difference sets. Given $|S_A|$ and $|S_B|$, and a bound $\overline{m}$ on the size of the symmetric difference, we can compute bounds on $m_A$ and $m_B$. In particular, $m_A \leq (\overline{m} + |S_A| - |S_B|)/2$ and $m_B \leq (\overline{m} - |S_A| + |S_B|)/2$, giving an overall bound $m_A + m_B \leq \overline{m}$. Thus, according to Theorem 4.1, we can recover $\chi_{S_A}(Z)/\chi_{S_B}(Z)$ using $\overline{m}$ sample points.

## 4.3 Choosing Sample Points

If a sample point $k$ is chosen that is in the intersection of $S_A$ and $S_B$, then the value of both the numerator and the denominator will be zero. As such, the value of $\chi_{S_A}(k)/\chi_{S_B}(k)$ is undefined and the corresponding sample point appears unusable. On the other hand,

$$\left. \frac{\chi_{S_A}(Z)}{\chi_{S_B}(Z)} \right|_{Z=k} \tag{10}$$

is well defined, because the problematic term $(Z - k)$ cancels out.

We can avoid this peculiarity altogether by choosing sample points outside of $S_A \cap S_B$. However, restricting the sample space is not strictly necessary. Note that if $k$ is a zero of the square-free polynomial $p(Z)$, then it is neither a zero nor a singular point of $p(Z)/(Z - k)$. Thus, if $k$ is in $S_A \cap S_B$, the ratio

$$\frac{\chi_{S_A}(Z)/(Z-k)|_{Z=k}}{\chi_{S_B}(Z)/(Z-k)|_{Z=k}} \tag{11}$$

is well defined, and is in fact equal to Equation (10).

Thus, instead of simply computing the value of a polynomial $p(k)$ at sample point $k$, we will compute a *tagged value*: either $p(k)$, if $p(k) \neq 0$, or $p(k)/(Z-k)|_{Z=k}$ otherwise. The tag corresponding to the value will mark whether or not $p(k) = 0$ so that the other host knows which computation was performed. The key advantage of using tagged values as opposed to picking sample points over a constrained range is that $A$ and $B$ can agree upon sample points *a priori*, and thus the sample points themselves do not need to be communicated.

### 4.4 A Complete Protocol

Protocol 3 shows a complete protocol that integrates the techniques of Sections 4.2 and 4.3. Notice that Protocol 3 can be adapted to use over a broadcast channel, so that $A$ could broadcast a single message to a collection of hosts $\{B_i\}$, and every $B_i$ whose set $S_{B_i}$ did not differ by more than $\overline{m}$ from $S_A$ could compute $S_A \cup S_{B_i}$.

---

**Protocol 3** Set reconciliation protocol given upper bound on $m$

---

Assume that $\overline{m}$ is an upper bound on $m$, the size of the symmetric difference between $S_A$ and $S_B$.

1. Host $A$ computes the tagged values of $\chi_{S_A}(Z)$ at points $\{0, 1, 2, \ldots, \overline{m}\}$. Host $A$ sends those values along with $\overline{m}$ and $|S_A|$ to host $B$.
2. Host $B$ computes the tagged values of $\chi_{S_B}(Z)$ at points $\{0, 1, 2, \ldots, \overline{m}\}$ and the corresponding values of $\chi_{S_A}(Z)/\chi_{S_B}(Z)$.
3. Host $B$ interpolates the unique rational function $f(Z)$ that matches the ratios in step 2 and whose numerator and denominator have degrees no greater than $(\overline{m} + |S_A| - |S_B|)/2$ and $(\overline{m} - |S_A| + |S_B|)/2$ respectively.
4. Host $B$ finds the roots of the numerator and denominator of $f(Z)$, which are the elements of $\Delta_A$ and $\Delta_B$ respectively.
5. Host $B$ sends the elements of $\Delta_B$ to $A$.

---

The following example demonstrates Protocol 3 concretely.

**Example 2** *Assume that $S_A = \{1, 9, 28, 33, 53, 61\}$ and $S_B = \{1, 9, 10, 28, 53\}$, that $\overline{m} = 3$ is an upper bound on the size of the symmetric difference between $S_A$ and $S_B$, and that all computations are done over the finite field $\mathbb{F}_{97}$. The protocol proceeds as follows:*

1. *Host $A$ computes the tagged values of its characteristic polynomial*

$$\chi_{S_A}(Z) = (Z-1)(Z-9)(Z-28)(Z-33)(Z-53)(Z-61)$$

*at sample points $\{0, 1, 2, 3\}$. Note that we use* `zero` *and* `nonzero` *to tag the values.*

$$\begin{array}{rcrcr} n_0 & = & \chi_{S_A}(Z)|_{Z=0} & \equiv & (\texttt{nonzero},\ 41) \\ n_1 & = & \chi_{S_A}(Z)/(Z-1)|_{Z=1} & \equiv & (\texttt{zero},\ 85) \\ n_2 & = & \chi_{S_A}(Z)|_{Z=2} & \equiv & (\texttt{nonzero},\ 65) \\ n_3 & = & \chi_{S_A}(Z)|_{Z=3} & \equiv & (\texttt{nonzero},\ 81) \end{array}$$

*The value $n_1$ is computed differently because $\chi_{S_A}(Z)|_{Z=1} = 0$, so it is correspondingly tagged. Host $A$ then sends its tagged values to host $B$.*

2. *Host $B$ likewise computes the tagged values of its characteristic polynomial*

$$\chi_{S_B}(Z) = (Z-1)(Z-9)(Z-10)(Z-28)(Z-53):$$

$$\begin{array}{rcrcr} d_0 & = & \chi_{S_B}(Z)|_{Z=0} & \equiv & (\texttt{nonzero},\ 9) \\ d_1 & = & \chi_{S_B}(Z)/(Z-1)|_{Z=1} & \equiv & (\texttt{zero},\ 14) \\ d_2 & = & \chi_{S_B}(Z)|_{Z=2} & \equiv & (\texttt{nonzero},\ 51) \\ d_3 & = & \chi_{S_B}(Z)|_{Z=3} & \equiv & (\texttt{nonzero},\ 46) \end{array}$$

3. *Host B interpolates the unique monic rational function $f(Z)$ that matches*

$$\begin{array}{ccccccc} f(0) & = & n_0/d_0 & = & 41/9 & \equiv & 80 \\ f(1) & = & n_1/d_1 & = & 85/14 & \equiv & 13 \\ f(2) & = & n_2/d_2 & = & 65/51 & \equiv & 26 \\ f(3) & = & n_3/d_3 & = & 81/46 & \equiv & 84 \end{array}$$

   *and has numerator and denominator of degrees at most $m_A \leq 2$ and $m_B \leq 1$ respectively. The unique monic rational function satisfying these conditions is:*

$$f(Z) = \frac{Z^2 - 94Z + 73}{Z - 10} \tag{12}$$

4. *The roots of the numerator of $f(Z)$ (i.e. 33 and 61) are the elements of $S_A \setminus S_B$, and the trivial root of the denominator (i.e. 10) is the element of $S_B \setminus S_A$.*

5. *Host B sends the the missing element* 10 *to host A.*

For a full reconciliation, Protocol 3 needs to communicate $\overline{m}$ tagged values and $m_B$ missing bitstrings, for a total of

$$\overline{m}(\lceil \lg(q) \rceil + 1) + m_B b \leq (\overline{m} + m_B)b + 2\overline{m} \text{ bits.}$$

If $\overline{m}$ is chosen near $m = m_A + m_B$, then this is just over twice the information-theoretic bound in Section 2. 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 $O(|S|\overline{m})$. Note, however, that the value of the sample points can be maintained incrementally as the set $S_A$ is built up, with computational cost of $O(\overline{m})$ per insertion and deletion; moreover, the computation of the sample values does not have to be redone for every run of the reconciliation protocol. The cost of interpolation and factoring is $O(\overline{m}^3 \log q)$, as illustrated in the appendix, although, as before, faster asymptotic times can be achieved with more sophisticated interpolation and factoring algorithms.

## 4.5 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, we need some way of detecting that enough samples have been taken to recover the rational function $\chi_{\Delta_A}(Z)/\chi_{\Delta_B}(Z)$. The following theorem suggests one way of doing this.

**Theorem 4.2** *Let $f(Z)$ and $g(Z)$ be distinct monic rational functions over $\mathbb{F}_q$ whose numerator and denominator have degree no more than $m_1$ and $m_2$ respectively. If $k$ is a randomly chosen element of $\mathbb{F}_q$, then the probability that $f(Z)$ and $g(Z)$ take on the same values at $Z = k$ is less than or equal to $(m_1 + m_2 - 1)/q$.*

This theorem follows from the fact that distinct monic rational functions with the given degree bounds cannot agree on more than $m_1+m_2-1$ points, which follows from Theorem 4.1. Theorem 4.2 can be used to test probabilistically whether an interpolated rational function $g(Z)$ is in fact equal to $f(Z) = \chi_{\Delta_A}(Z)/\chi_{\Delta_B}(Z)$.

To implement such a test, note that $|S_A|$ and $|S_B|$ are trivial upper bounds on $m_A$ and $m_B$ respectively. Let $g(Z)$ be a rational function reconstructed from $m'$ sampled values of $f(Z)$. If $g(Z)$ is not equal to $f(Z)$, then, by Theorem 4.2, the probability that they agree at a randomly selected evaluation point is no greater than $\rho = (|S_A| + |S_B| - 1)/q$. By using enough random evaluation points, $B$ can determine that it has reconstructed $g(Z) = f(Z)$ with probability $1 - \epsilon$, for some $\epsilon > 0$. Note that $\rho$ is typically quite small, so few evaluation points are needed.

Note that this approach 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 in place of all the sample points.

There is also the question of how sample points are transmitted. One approach would be for $A$ to send each sample point sequentially, until $B$ determines with a sufficient degree of certainty that the correct rational function has been recovered. The drawback to this approach is that if $r > m$ samples are needed, then the protocol will require $r$ rounds. Alternately, $A$ could double the number of sample points each round. This would require only $\log(r)$ rounds, at the cost of sending up to $r$ extra sampled values.

## 5 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.

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 [14]. The relationship between secret sharing and set reconciliation deserves further study.

These protocols have been implemented using Victor Shoup's well known NTL package[15]. 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.

## 6 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.

## References

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

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

[3] Demers, A. J., Greene, D. H., Hause, C., Irish, W., and Larson, J. 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), no. 6, ACM, pp. 1–12.

[4] Evfimievski, A. A probabilistic algorithm for updating files over a communication link. *Theoretical Computer Science* (2000), 191–199.

[5] Golding, R. A. *Weak-Consistency Group Communication and Membership*. PhD thesis, UC Santa Cruz, December 1992. Published as technical report UCSC-CRL-92-52.

[6] Guo, K., Hayden, M., Renesse, R. v., Vogels, W., and Birman, K. P. Gsgc: An efficient gossip-style garbage collection scheme for scalable reliable multicast. Tech. rep., Cornell University, December 1997.

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

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

[9] Hayden, M., and Birman, K. Probabilistic broadcast. Tech. rep., Cornell University, 1996.

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

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

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

[13] Orlitsky, A. Interactive communication of balanced distributions and correlated files. *SIAM Journal on Discrete Mathematics 6*, 4 (November 1993), 548–564.

[14] Shamir, A. How to share a secret. *CACM 22* (Nov. 1979), 612–613.

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

[16] Shoup, V. Factoring polynomials over finite fields: Asymptotic complexity vs. reality. In *Proc. IMACS Symposium* (Lille, France, 1993).

[17] Shoup, V. A new polynomial factorization algorithm and its implementation. *Journal of Symbolic Computation 20* (1995), 363–397.

[18] Stoer, J., and Bulirsch, R. *Introduction to Numerical Analysis*, 2nd ed. Springer-Verlag, New York, 1993.

[19] Trachtenberg, A., and Minsky, Y. Efficient reconciliation of unordered databases. Tech. rep., Cornell University, 1999.

[20] van Renesse, R. Captain cook: A scalable navigation service. In preparation.

[21] van Renesse, R., Minsky, Y., and Hayden, M. A gossip-style failure detection service. In *Middleware '98: IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing* (1998), N. Davies, K. Raymond, and J. Seitz, Eds., Springer Verlag, pp. 55–70.

[22] Zippel, R. E. *Effective Polynomial Computation.* Kluwer Academic Press, Boston, 1993.

## 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 $O(d^{1.82} \lg q)$ [10], 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, 16, 17, 22]). 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 $O(d^2 \log q)$ by using repeated squaring to compute $h(Z) = Z^q \pmod{f(Z)}$ and then testing if $h(Z) - Z \equiv 0 \pmod{f(Z)}$. The repeated squaring stage dominates the time complexity, since $O(\log p)$ 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, $O(d^2 \log p)$ 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 Rational Interpolation

Let $f(Z)$ be a rational function 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 $p_0 = q_0 = 1$. The *degree* of $f(Z)$ is $\max(m,n)$ and 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 $P_1Q_2 = P_2Q_1$. That is, if one rational function can be reduced to the other 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 $(k_i, f_i) \in V$.

The problem of determining a rational function with bounded degrees that satisfies a support set is called the *rational interpolation problem.* Most work on this problem is for rational functions with floating point coefficients, where accuracy and numerical stability are an issue. There are two assumptions that distinguish 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.

The first proposition implies that if there is a rational function corresponding to the given support set,[1] then there is only one such rational function. The following theorem is an adaptation of a standard theorem on rational interpolation (see [18], Proposition 2.2.1.4), and parallels Theorem 4.1.

**Theorem B.1 (Interpolation of Rational Functions)** *Let $V$ be a support set with $m_A + m_B$ elements over a field $K$. Assume there exist two monic rational functions with degrees $(m_A, m_B)$ that satisfy the support set. Then the two rational functions are equivalent.*

**Proof:** Denote the two 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 the degree of $P(Z)$ is

[1]Not all support sets have corresponding rational functions.

$m_A + m_B - 1$ and it vanishes at $m_A + m_B$ points, $P(Z)$ must be identically zero. ∎

Assume we have a support set $V = \{(h_1, f_1), \dots, (h_m, f_m)\}$ and that the rational function $P(Z)/Q(Z)$ satisfies $V$. Further assume, that $P$ and $Q$ are monic with the structure below:

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

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

$$k_i^{m_A} + p_1 k_i^{m_A - 1} + \cdots + p_{m_A} = f_i(k_i^{m_B} + q_1 k_i^{m_B - 1} + \cdots + q_{m_B}).$$

We may combine $m$ of these relations to form a *Generalized Vandermonde* system of equations:

$$\begin{pmatrix} k_1^{m_A-1} & \cdots & k_1 & 1 & -f_1 k_1^{m_B-1} & \cdots & -f_1 k_1 & -f_1 \\ k_2^{m_A-1} & \cdots & k_2 & 1 & -f_2 k_2^{m_B-1} & \cdots & -f_2 k_2 & -f_2 \\ \vdots & & & & \vdots & & & \vdots \\ k_m^{m_A-1} & \cdots & k_m & 1 & -f_m k_m^{m_B-1} & \cdots & -f_m k_m & -f_m \end{pmatrix} \cdot \begin{pmatrix} p_1 \\ \vdots \\ p_{m_A} \\ q_1 \\ \vdots \\ q_{m_B} \end{pmatrix} = \begin{pmatrix} f_1 k_1^{m_B} - k_1^{m_A} \\ f_2 k_2^{m_B} - k_2^{m_A} \\ \vdots \\ f_m k_m^{m_B} - k_m^{m_A} \end{pmatrix}. \tag{13}$$

We denote this system of equations by $\mathcal{S}(m_A, m_B; V)$. Being a solution of Equation (13) is a necessary condition for a monic rational function to satisfy the support set $\{(k_1, f_1), \dots, (k_m, f_m)\}$.

Assume $f(Z) = P(Z)/Q(Z)$ is a monic, reduced rational function of degrees $(m_A, m_B)$ that satisfies the support set $V$ and that has $m_A + m_B$ elements. By Theorem 4.1, $f(Z)$ is the only monic reduced rational function of degrees $(m_A, m_B)$ that satisfies Equation (13). Furthermore, the system $\mathcal{S}(m_A + \ell, m_B + \ell; V)$ is singular since the coefficients of $P(Z) \cdot s_\ell(Z)$ and $Q(Z) \cdot s_\ell(Z)$ are solutions whenever $s_\ell(Z)$ is a monic polynomial of degree $\ell$. There cannot be any other solutions because that would lead to two inequivalent solutions of Equation (13) of degrees $(m_A + \ell, m_B + \ell)$, which would contradict Theorem 4.1. This system of equations can be solved using classical Gaussian elimination in $O(m^3)$ operations.

To illustrate some of the issues that can arise when solving this system of equations consider the support set:

$$V = \{(1, \frac{2}{3}), (2, \frac{3}{4}), (3, \frac{4}{5}), (4, \frac{5}{6}), \dots\}.$$

Using the first two values gives the linear equations

$$\begin{pmatrix} 1 & -\frac{2}{3} \\ 1 & -\frac{3}{4} \end{pmatrix} \cdot \begin{pmatrix} a_1 \\ b_1 \end{pmatrix} = \begin{pmatrix} \frac{1}{2} \\ \frac{3}{4} \end{pmatrix}$$

which is easily solved $a_1 = 1$, $b_1 = 2$, to yield the rational function

$$\frac{Z+1}{Z+2}.$$

This low degree rational function actually satisfies the entire support set $V$. All other degree 1 rational functions that satisfy $V$ are of the form

$$\frac{aZ + a}{aZ + 2a}.$$

The monotonicity assumption eliminates this possibility and thus the corresponding system of equations in Equation (13) has a unique solution.

If higher degree rational functions are used then the system of equations becomes singular. This is because the corresponding higher degree rational function will necessarily have additional factors in the numerator and denominator. This is illustrated below, where we obtain a rational function whose numerator and denominator are both of degree three.

If $P$ and $Q$ are both degree 3, then we get the following system of linear equations

$$\begin{pmatrix} 1 & 1 & 1 & -\frac{2}{3} & -\frac{2}{3} & -\frac{2}{3} \\ 4 & 2 & 1 & -3 & -\frac{3}{2} & -\frac{3}{4} \\ 9 & 3 & 1 & -\frac{36}{5} & -\frac{4}{5} & -\frac{2}{5} \\ 16 & 4 & 1 & -\frac{40}{3} & -\frac{10}{3} & -\frac{5}{6} \\ 25 & 5 & 1 & -\frac{150}{7} & -\frac{30}{7} & -\frac{6}{7} \\ 36 & 6 & 1 & -\frac{63}{2} & -\frac{21}{4} & -\frac{7}{8} \end{pmatrix} \cdot \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} \frac{1}{3} \\ 2 \\ \frac{27}{5} \\ \frac{32}{3} \\ \frac{125}{7} \\ 27 \end{pmatrix} \tag{14}$$

Though the system is singular, but a basis for all of the solutions can be found using any two of $\{a_1, a_2, a_3, b_1, b_2, b_3\}$. Using $a_2$ and $a_3$ as the basis gives a solution of:

$$\begin{aligned} a_1 &= 1 + a_2 - a_3, & b_2 &= 2a_2 - a_3, \\ b_1 &= 2 + a_2 - a_3, & b_3 &= 2a_3. \end{aligned}$$

For any choice of $a_2$ and $a_3$ these values will be a solution of (14). Thus, the rational function corresponding to the set of equations (14) is

$$\frac{Z^3 + (1 + a_2 - a_3)Z^2 + a_2 Z + a_3}{Z^3 + (2 + a_2 - a_3)Z^2 + (2a_2 - a_3)Z + 2a_3} = \frac{Z+1}{Z+2} \cdot \frac{Z^2 + (a_2 - a_3)Z + a_3}{Z^2 + (a_2 - a_3)Z + a_3} = \frac{Z+1}{Z+2},$$

clearing the common factor from the numerator and denominator. So each of the infinite number of solutions of (14) leads to the same reduced rational functions $(Z + 1)/(Z + 2)$.

In Sections 4.3 and 4.4 tagged values are used to eliminate restrictions on the choice of evaluation points. In this case, the evaluation points may lead to a zero in the numerator, denominator, or both. Formulating the rational interpolation problem using points in projective space deals with these issues most cleanly.

The *symmetric closure* of a field $K$ is a space $K \times K$ where two elements $(r_1, s_1)$ and $(r_2, s_2)$ are equivalent if there exists $\ell$, is a non-zero element of $K$, such that $r_1\ell = r_2$ and $s_1\ell = s_2$. The equivalence class corresponding to $(r, s)$ is denoted by $\langle r, s\rangle$. The symmetric closure of $K$ is denoted by $\hat{K}$. $K$ can be embedded in $\hat{K}$ by $r \mapsto \langle r, 1\rangle$, and a point $\langle r, s\rangle$ corresponds to $r/s \in K$ unless $s = 0$.

A *projective support set* $V$ is a set of pairs $(k_i, \langle r_i, s_i\rangle) \in K \times \hat{K}$ where the $k_i$ are distinct. A rational function $P(Z)/Q(Z)$. satisfies $V$ if

$$P(k_i)s_i - Q(k_i)r_i = 0$$

for all $(k_i, \langle r_i, s_i\rangle) \in V$.

Using projective support sets, the system of linear equation generated will be of the form

$$\begin{pmatrix} s_1 k_1^{m_A-1} & \cdots & s_1 & -r_1 k_1^{m_B-1} & \cdots & -r_1 & \\ s_2 k_2^{m_A-1} & \cdots & s_2 & -r_2 k_2^{m_B-1} & \cdots & -r_2 & \\ \vdots & & & & & \vdots & \vdots \\ s_1 k_1^{m_A-1} & \cdots & s_1 & -r_1 k_1^{m_B-1} & \cdots & -r_1 & \\ s_m k_m^{m_A-1} & \cdots & s_m & -r_m k_m^{m_B-1} & \cdots & -r_m & \end{pmatrix} \cdot \begin{pmatrix} p_1 \\ \vdots \\ p_{m_A} \\ q_1 \\ \vdots \\ q_{m_B} \end{pmatrix} = \begin{pmatrix} r_1 k_1^{m_B} - s_1 k_1^{m_A} \\ r_2 k_1^{m_B} - s_2 k_1^{m_A} \\ \vdots \\ r_m k_1^{m_B} - s_m k_1^{m_A} \end{pmatrix}.$$

At each evaluation point $k_i$ the technique of Section 4.3 will yield two typed points $(k_i, (\texttt{Atag}_i, r_i))$ and $(k_i, (\texttt{Btag}_i, s_i))$. If $\texttt{Atag}_i$ is equal to $\texttt{Btag}_i$ then we used a projective support element of $(k_i, \langle r_i, s_i \rangle)$. If they are different then $(k_i, \langle 0, s_i \rangle)$ is used if $\texttt{Atag}_i$ is equal to `zero` and $(k_i, \langle r_i, 0 \rangle)$ if $\texttt{Btag}_i$ is equal to `zero`.