Efficient Traitor Tracing Algorithms Using List Decoding

Alice Silverberg[1⋆], Jessica Staddon[2⋆⋆], and Judy L. Walker[3⋆⋆⋆]

[1] Department of Mathematics
Ohio State University
Columbus, OH, USA
silver@math.ohio-state.edu

[2] Xerox PARC
Palo Alto, CA, USA
jstaddon@parc.xerox.com

[3] Department of Mathematics and Statistics
University of Nebraska
Lincoln, NE, USA
jwalker@math.unl.edu

Abstract. We use powerful new techniques for list decoding error-correcting codes to efficiently trace traitors. Although much work has focused on constructing traceability schemes, the complexity of the tracing algorithm has received little attention. Because the TA tracing algorithm has a runtime of $O(N)$ in general, where N is the number of users, it is inefficient for large populations. We produce schemes for which the TA algorithm is very fast. The IPP tracing algorithm, though less efficient, can list all coalitions capable of constructing a given pirate. We give evidence that when using an algebraic structure, the ability to trace with the IPP algorithm implies the ability to trace with the TA algorithm. We also construct schemes with an algorithm that finds all possible traitor coalitions faster than the IPP algorithm. Finally, we suggest uses for other decoding techniques in the presence of additional information about traitor behavior.

1 Introduction

Traceability schemes are introduced in [9] and have been extensively studied in the intervening years for use as a piracy deterrent. We focus on one of the few aspects of this area of work that has received little attention: the complexity

⋆ Silverberg would like to thank MSRI, Bell Labs Research Silicon Valley, NSA, and NSF.

⋆⋆ Much of this work was completed while Staddon was employed by Bell Labs Research Silicon Valley.

⋆⋆⋆ Walker is partially supported by NSF grants DMS-0071008 and DMS-0071011.

C. Boyd (Ed.): ASIACRYPT 2001, LNCS 2248, pp. 175–192, 2001.

of the traitor tracing algorithms. We show that powerful new techniques for the list decoding of error-correcting codes enable us to construct traceability schemes with very fast traitor tracing algorithms. Further, we use list decoding to give new algorithms for producing a list of all coalitions capable of creating a given pirate. In addition, we discuss potential applications of other decoding methods to the problem of tracing traitors, suggest alternative approaches when additional information is known about the way the traitors are operating, and examine the relationship between two important tracing algorithms.

In a popular model for traceability schemes a unique set (possibly ordered) of r symbols is associated with each user. For example, the set may be associated with a user's software CD, or contained in a smartcard the user has for the purpose of viewing encrypted pay-TV programs (in the latter case, the set corresponds to a set of keys). When a coalition forms to commit piracy, it must construct a set to associate with the pirate object. In the case of unordered sets, this pirate set consists of r symbols, each of which belongs to at least one coalition member's set. If the sets are ordered, the coalition members must form an ordered pirate set in which the symbol in each position is identical to the symbol in the same position in the ordered set of some coalition member. In either scenario a traitor tracing algorithm is applied to the pirate, and identifies an actual traitor or traitors. The approach we take here is to use error-correcting codes to construct traceability schemes in which the sets are ordered. The ordered (as opposed to the unordered) set scenario yields naturally to coding theoretic techniques and has many practical applications ([10,7]).

We first focus on the TA traitor tracing algorithm (following the terminology in [40]), that identifies as traitors all users who share the most with the pirate. In general the TA algorithm runs in $O(N)$ time, where N is the number of users. However, this paper shows that for suitable constructions based on error-correcting codes, tracing can be accomplished in time polynomial in $c \log N$, where c is the maximum coalition size. This is a significant improvement, as we expect c to be much smaller than N. The constructions in this paper match the best previously known schemes in this model in terms of the alphabet size that is required to achieve a certain level of traceability for a given codeword length, and exceed all earlier schemes in the speed with which they trace (at least) one traitor.

We also consider the IPP tracing algorithm (following the terminology in [23]). The IPP algorithm identifies all coalitions capable of making a pirate and looks for a common member(s) amongst these coalitions. Hence, the IPP property seems to be a more fundamental traceability property. In general this algorithm runs in time $O(crN^c)$, where r is the length of each codeword, and hence is even less efficient than the TA algorithm. However, there are two good reasons to be interested in IPP codes. First, the extra computational burden of the IPP algorithm has led to the question (see [37]) of whether IPP schemes may beat TA schemes in other respects, namely, in terms of the number of codewords for a fixed set of parameters. We provide evidence that for schemes with enough structure to enable efficient tracing algorithms, increasing the number of

codewords causes tracing to fail with *both* the TA and IPP algorithms. Hence, IPP codes do not appear to yield efficiency improvements in this respect. Secondly, as part of the IPP tracing process, additional valuable piracy information is amassed, namely, a list of all coalitions capable of creating the pirate in question. Such a list is not a by-product of the TA algorithm, but is a useful part of a security audit. We show that when error-correcting codes are used to construct TA traceability codes (which are also IPP codes, by a result in [37]), list decoding techniques can be used to construct new algorithms for finding all such coalitions. We give an algorithm that is more efficient than the brute force approach of the IPP algorithm of evaluating each coalition for its ability to create the pirate, thereby answering an open question in [37].

This paper gives the first applications of list decoding to the traitor tracing problem in the above model, although Zane [48] uses such techniques to address the related problem of watermark detection. (See Section 1.1 below for a discussion of this, and other, related work.) These list decoding techniques are receiving wide attention in the coding theory community, and improvements and generalizations are being rapidly produced. We believe that in this paper we have merely scratched the surface of the potential applications of decoding techniques to traceability. In the last section we discuss the use of other decoding methods when additional information is known about the traitors or how they operate, giving directions for future work in this area.

Overview. Section 1.1 covers related work on traceability and broadcast encryption and Section 2 covers the necessary background on traceability and coding theory. Section 3 describes how to construct efficient traceability schemes. Section 4 considers the relationship between TA and IPP traceability schemes, providing justification for our restriction to the TA case, and raising some questions concerning the relationship between TA and IPP for linear codes. Section 5 shows how codes of sufficiently large minimum distance enable a more efficient algorithm for finding all coalitions of traitors. A discussion of other potential applications of coding theoretic ideas and techniques to traceability questions is given in Section 6.

1.1 Related Work

The phrase *traitor tracing* is coined in [9] (see also the extended version [10]). In traceability schemes, users are each given an ordered (as in [9,7,15,37], for example), or unordered (as in [40], for example) set of keys.

In [6] (see also the revised version [7]), methods for creating TA traceability codes are given for the purpose of fingerprinting digital data. Lower bounds and additional constructions of TA traceability schemes are given in [40], while lower bounds are also proven in [27,26]. In addition, [26] provides a tracing algorithm for schemes in [27].

The problem of combining broadcast encryption and traceability is studied in [41,16,29,46].

Variations on the models of [10,7] have been studied in recent years. *Dynamic* models (here we study a static model), in which it is possible to get additional evidence of piracy in order to "test" traitor guesses, are studied in [15,3,33]. A public-key traitor tracing scheme is given in [5]. One of the nice properties of the scheme in [5] is that it is possible to identify *all* traitors. We note that although our algorithms in Sect. 3 can only guarantee the identification of one traitor, they do so in significantly faster time (polynomial in $c \log N$, versus $O(N\log^2 N\log\log N)$ in [5], with N the number of codewords and c the maximum coalition size).

In [31,11], ways in which accountability can be added to the model are discussed. For example, to improve upon the strength of the deterrent, in [11] committing piracy efficiently necessitates revealing sensitive information. In [17], a system in which pirate pay-TV decoders can only work for short periods of time is presented. As noted in [17], traceability can be a useful addition to a long-lived broadcast encryption scheme. If keys are allocated to smartcards in such a way as to ensure some traceability, it is possible to keep a list of traitor smartcards over time. If the smartcard of one particular user appears on the list frequently despite many smartcard refreshments (i.e., key changes) this mounting evidence makes it increasingly likely that the user is actually guilty, and not simply a victim of smartcard theft. Hence, as long as traceability schemes are efficient, they can quickly yield useful information during system audits.

Recently, the identifiable parent property (IPP) tracing algorithm has garnered attention [23,2,37] (also, very similar ideas are studied in [39]). In [23], a combinatorial characterization of 2-IPP schemes is presented. Additional constructions of and bounds for IPP schemes appear in [2,37].

A coding theoretic approach is taken in [25] to study the related problem of blacklisting users in a broadcast encryption scheme, but that paper does not address the question of tracing.

Our approach takes advantage of recent powerful list decoding methods, which originated with the work of Sudan [42]. In list decoding the input is a received word and the output is the list of all codewords within a given Hamming distance of the received word. Sudan's results by themselves are not strong enough to be applicable in the setting in which the TA algorithm succeeds in finding traitors (as opposed to identifying probable traitors), since the decoding procedure in [42] is not capable of correcting enough errors in the code. However, Sudan's work has recently been extended to enable it to efficiently correct more errors; i.e., it extends the radius of the Hamming ball around the received word in which it can find all the codewords in time polynomial in the length of the codewords. The improvements in [19] are precisely sufficient to be applicable to the setting where the TA algorithm succeeds. An additional advantage of this method is that it gives a list containing one or more traitors, rather than only one. Efficient list decoding algorithms now exist for Reed-Solomon codes, more general algebraic geometry codes, and some concatenated codes.

List decoding techniques are applied to the problem of watermarking in [48]. Whereas in traceability schemes each user has a unique codeword, in the wa-

termarking scenario each user needs to be given the same "document" V, taken in [48] to be a vector of real numbers between 0 and 1. To prevent users from distributing pirated copies of V, each user is given a distinct, slightly modified "watermarked" version of V. The CKLS media watermarking scheme [8] is modified in [48] so that the watermarks are chosen from a set of randomly generated CKLS codes according to a Reed-Solomon code. Given a suspected pirate copy of V, the results of [42] on list decoding can then be used to identify one or more traitors.

Here, we consider the related question of traceability schemes, and we apply list decoding results for algebraic geometry codes and certain concatenated codes in addition to Reed-Solomon codes. In [48], Reed-Solomon codes are used to obtain vectors of real numbers between 0 and 1 to serve as a watermark, while here the error-correcting codes themselves are the traceability schemes.

We note that algebraic geometry codes appear to have been under-utilized in cryptological applications. For example, the results of [34] can be used to give better explicit examples of c-frameproof codes than those obtained in [7]. The codes constructed in [34] are concatenated codes (see below) where the outer code is an algebraic geometry code coming from a Hermitian curve, while those used in [7] come from pseudo-random graphs (see [1]).

2 Background on Codes and Traceability

In this section we give definitions, notation, and background on codes, traceability, and the decoding techniques that form the basis for our tracing algorithms.

2.1 Definitions and Notation

A *code* C of *length* r is a subset of Q^r, where Q is a finite alphabet. The elements of C are called *codewords*; each codeword has the form $x = (x_1, \cdots, x_r)$, where $x_i \in Q$ for $1 \leq i \leq r$. Subsets of C will be called *coalitions*.

For any coalition $C_0 \subseteq C$, we define the set of *descendants* of C_0, denoted $\mathsf{desc}(C_0)$ by

$$\mathsf{desc}(C_0) = \{w \in Q^r : w_i \in \{x_i : x \in C_0\}, \text{ for all } 1 \leq i \leq r\} \ .$$

The set $\mathsf{desc}(C_0)$ consists of the r-tuples that could be produced by the coalition C_0.

We define $\mathsf{desc}_c(C)$ to be the set of all $x \in Q^r$ for which there exists a coalition C_0 of size at most c such that $x \in \mathsf{desc}(C_0)$. In other words, $\mathsf{desc}_c(C)$ consists of the r-tuples that could be produced by a coalition of size at most c.

For $x, y \in Q^r$, let $I(x, y) = \{i : x_i = y_i\}$.

Definition 1. *A code C is a c-TA (traceability) code if for all coalitions C_i of size at most c, if $w \in \mathsf{desc}(C_i)$ then there exists $x \in C_i$ such that $|I(x, w)| > |I(z, w)|$ for all $z \in C - C_i$.*

In other words, C is a c-TA code if, whenever a coalition of size at most c produces a pirate word w, there is an element of the coalition which is closer to w than any codeword not in the coalition.

Codes with the identifiable parent property (IPP) are another type of traceability code.

Definition 2. *A code C is a c-IPP code if for all $w \in \mathsf{desc}_c(C)$, the intersection of the coalitions C_i of size at most c such that $w \in \mathsf{desc}(C_i)$ is nonempty.*

Suppose C is a code of length r. The *(Hamming) distance* between two elements x and y of Q^r is $r - |I(x, y)|$. The *minimum distance* of the code C is the smallest distance between distinct codewords of C.

If C is a c-IPP code and $w \in \mathsf{desc}_c(C)$, then the *traitors* that can produce the *pirate* w are the codewords that lie in all coalitions C_i of size at most c such that $w \in \mathsf{desc}(C_i)$.

When implementing one of the traceability codes just described, one randomly chooses a set of symbols $\{s_{(i,y)}\}$ with $i \in \{1, \ldots, r\}$ and y in the alphabet Q, and the collection of symbols corresponding to a given user is determined by the codeword associated with that user. For example, if the codeword $x = (x_1, \ldots, x_r)$ is associated with user u, then the set of symbols associated with user u is $S_u = \{s_{(1,x_1)}, \ldots, s_{(r,x_r)}\}$. It is S_u, not x, that the user stores (e.g., S_u is embedded in the user's CD or smartcard). The encryption step makes the model of pirate behavior that we consider reasonable. Since the symbols are generated randomly it is essentially impossible to guess a symbol, and hence a coalition is only able to form a pirate out of its pooled collection of symbols. In other words, moving from codewords to symbols thwarts algebraic attacks (such as, for example, the attack on [27] found in [41,5]). Although a coalition may be able to write down any codeword (this information may be public), it can only generate the symbol associated with an entry in the codeword if there is a coalition member that agrees with the codeword in that position.

2.2 Background Traceability Results

The following result, which is Lemma 1.3 of [37], is very useful for showing that a code is c-IPP.

Lemma 1. ([37], Lemma 1.3) *Every c-TA code is a c-IPP code.*

As shown in [37], there are c-IPP codes that are not c-TA. We give a simple example of a 2-IPP code that is not 2-TA.

Example 1. Let $u_1 = (0, 0, 1)$, $u_2 = (1, 0, 0)$, and $u_3 = (2, 0, 0)$. The code $\{u_1, u_2, u_3\}$ is clearly 2-IPP, since the first entry of a pirate determines a traitor. The coalition $\{u_1, u_2\}$ can produce the pirate $w = (0, 0, 0)$. However, $|I(u_1, w)| = |I(u_2, w)| = |I(u_3, w)| = 2$, so the code is not 2-TA.

Note that for c-IPP codes, traitor tracing is roughly an $O(\binom{N}{c})$ process, where N is the total number of codewords in the code. A traitor tracing algorithm for a c-TA code takes as input a $w \in \mathsf{desc}_c(C)$ and outputs a codeword x such that $|I(x,w)|$ is largest. Hence for c-TA codes, tracing is an $O(N)$ process, in general.

The next result, which is proved in [37] (see Theorem 4.4 of that paper; see also [9] and [10]), shows that for codes with large enough minimum distance the TA algorithm suffices, and consists of finding codewords within distance $r - \frac{r}{c}$ from the pirate. Further, all codewords within this distance will be traitors.

Theorem 1. ([37], Theorem 4.4) *Suppose C is a code of length r, c is a positive integer, and the minimum distance d of C satisfies $d > r - \frac{r}{c^2}$. Then*

(i) C is a c-TA code;
(ii) if C_0 is a coalition of size at most c, and $w \in \mathsf{desc}(C_0)$, then:
  (a) there exists an element of C_0 within distance $r - \frac{r}{c}$ of w, and
  (b) every codeword within distance $r - \frac{r}{c}$ of w is in the coalition C_0.

2.3 Linear Codes

Linear codes are a very important class of codes. We will say that a code of length r is *linear*, or linear over F_q, if the alphabet is a finite field F_q and the code is a linear subspace of the vector space F_q^r. The *dimension* of the code is its dimension as a vector space. If C is a linear code over F_q of dimension k, then $|C| = q^k$.

Reed-Solomon codes are among the most widely-used linear codes, with many useful applications (e.g., compact disks). To obtain a Reed-Solomon code of length r and dimension k over the finite field F_q, fix r distinct elements $\alpha_1, \ldots, \alpha_r$ of F_q. The codewords are exactly the r-tuples $(f(\alpha_1), \ldots, f(\alpha_r))$ as f runs over (the zero polynomial and) all polynomials of degree less than k in $F_q[x]$. Note that a basis for the code over F_q is

$$\{(1, \ldots, 1), (\alpha_1, \ldots, \alpha_r), (\alpha_1^2, \ldots, \alpha_r^2), \ldots, (\alpha_1^{k-1}, \ldots, \alpha_r^{k-1})\} \ .$$

Since two distinct polynomials of degree less than k agree on at most $k-1$ points, the minimum distance of the code is $r - k + 1$.

A useful generalization of Reed-Solomon codes are *algebraic geometry (AG) codes* (see, for example, [18,38,44]). The linear codes with the "best" known parameters asymptotically are AG codes [45]. One advantage of AG codes is that they are not, in general, bound by the restriction that $r \leq q$, as was the case for the Reed-Solomon codes above. Being freed of this constraint allows us to have a smaller alphabet (and in applications, fewer keys), for given choices of the other parameters. Hermitian codes, coming from Hermitian curves, are examples of AG codes that have nice properties and can be defined explicitly. For those familiar with the below terminology (such knowledge is not essential for appreciating the results of this paper), we note that for our purposes it suffices to consider the one-point codes $C_X(\mathcal{P}, \ell P_0)$ which can be defined as follows. Start with a smooth, absolutely irreducible curve X of genus g defined over a

finite field F_q, a set $\mathcal{P} = \{P_1, \ldots, P_r\}$ of r distinct F_q-rational points on X, another F_q-rational point P_0 on X which is not in the set $\mathcal{P}$, and an integer ℓ. The codewords are then the r-tuples $(f(P_1), \ldots, f(P_r))$, where f runs over the rational functions on X whose only pole is P_0, where the multiplicity is at most ℓ. If $2g - 2 < \ell < r$, this code has dimension $\ell + 1 - g$ and minimum distance at least $r - \ell$. Reed-Solomon codes can be viewed as algebraic geometry codes by taking X to be the projective line, $\mathcal{P}$ to be the set of points corresponding to the r chosen field elements, P_0 to be the point at infinity, and $\ell = k - 1$.

Concatenated codes are codes which are "concatenated" from two other codes. When two linear codes are concatenated, the product of their lengths (resp., dimensions, resp., minimum distances) is the length (resp., dimension, resp., minimum distance) of the (linear) concatenated code. There are linear concatenated codes for small alphabets which have good list decoding capabilities, i.e., a small list of possible codewords can be recovered even when a large percentage of the symbols are in error or have been erased [20].

We refer the reader to [18,28,38,44] for more information on coding theory.

2.4 Decoding

In the theory of error-correcting codes, a codeword is transmitted through a noisy channel and an element of Q^r (i.e., a *word*) is received. The receiver (or *decoder*) then tries to determine as accurately as possible which codeword was transmitted.

If d is the minimum distance of the code, then the receiver can "correct" $t = \lfloor \frac{d-1}{2} \rfloor$ errors; i.e., there is at most one codeword within distance t of the received word. The radius t is called the *error-correction bound* or the *packing radius*. *Minimum-distance* (or *nearest-neighbor*) *decoding* finds the closest codeword to the received word. In practice, minimum-distance decoding is very slow. In *bounded-distance* decoding, the decoder finds a codeword within a specified distance of the received word, if one exists. In the bounded-distance decoding decision problem, the inputs are a linear code over a given finite field, a received word, and a specified distance t, and the output is a yes or no answer to the question of whether there is a codeword within distance t of the received word. This decision problem is known to be NP-complete [4].

In *list decoding*, the goal is to output the list of all codewords within a specified distance of the received word. In [42] and [43], Sudan gave the first efficient methods for list decoding that run in time polynomial in the length of the codewords. Since then, Sudan's list decoding technique has been improved, generalized, and refined [35,36,19,20,21,22,24,30,32,47,12,13]. The runtimes for the steps of the algorithm have been improved, the number of errors that can be "corrected" has been increased, and the technique has been shown to be applicable to a larger class of codes. Sudan's original algorithm is for Reed-Solomon codes. Other codes for which the techniques have been shown to apply include AG codes (for which the focus has been on Hermitian codes) and certain concatenated codes (see [20], where the "outer code" is a Reed-Solomon or AG code and the "inner code" is a Hadamard code).

In *erasure decoding*, some positions of the received word are garbled or "erased", and cannot be identified. In this case the decoder knows that errors occurred in those positions. In *erasure-and-error decoding*, the decoder receives a word with some erasures and some errors, and determines the transmitted word, or a list of possible transmitted words (given some appropriate bounds on the numbers of errors and erasures).

In *soft-decision decoding*, instead of receiving a (*hard-decision*) word, the decoder receives a reliability matrix that states the probability that any given element of the alphabet was sent in any given position. Using this "soft" information, a soft-decision decoder outputs the most likely transmitted codeword(s).

3 Efficient Tracing Algorithms via List Decoding

In this section we show how the efficiency of the TA tracing algorithm can be greatly improved when the traceability scheme is based on certain error-correcting codes, and the tracing algorithm uses fast list decoding methods. What is an $O(N)$ process in general becomes a process that runs in time polynomial in $c \log N$. These constructions match the best previously known traceability schemes in this model in terms of the alphabet size that is required to support a given level of traceability and codeword length (roughly speaking, the alphabet size is $O(N^{\frac{c^2}{r}})$). The following theorem describes constructions based on Reed-Solomon, algebraic geometry, and concatenated codes. One advantage of considering all three types of codes is that the appropriate code choice for the traceability scheme depends on the desired parameters.

Theorem 2. *(i) Let C be a Reed-Solomon code of length r and dimension k over a finite field F_q of size at most 2^r. If c is an integer, $c \geq 2$, and $r > c^2(k-1)$, then C is a c-TA code and there is a traitor tracing algorithm that runs in time $O(r^{15})$. If $r = (1+\delta)c^2(k-1)$ then the algorithm runs in time $O(\frac{r^3}{\delta^6})$. For $r = \Theta(c^2 k)$, the runtime is $O(c^{30} \log_q^{15} N)$.*

(ii) Let X be a nonsingular plane curve of genus g defined over a finite field F_q, $\mathcal{P}$ a set of r distinct F_q-rational points on X, P_0 an F_q-rational point on X which is not in $\mathcal{P}$, and k an integer such that $k > g-1$. Let c be an integer such that $c \geq 2$ and $r > c^2(k+g-1)$, assume that $q \leq 2^r$, and assume the pre-processing described in [19] has occurred. Then the one-point AG code $C_X(\mathcal{P}, (k+g-1)P_0)$ is a c-TA code with a traitor tracing algorithm that runs in time polynomial in r.

(iii) If k and c are positive integers, q is a prime power, $q > c^2 \geq 4$, and δ is a real number such that $0 < \delta \leq \frac{q/c^2-1}{q-1}$, then there exists an explicit linear c-TA code over the field F_q of length $r = O(\frac{k^2}{\delta^3 \log(1/\delta)})$ (or length $r = O(\frac{k}{\delta^2 \log^2(1/\delta)})$) and dimension k with a polynomial (in r) traitor tracing algorithm.

Proof. (i) Since C is a Reed-Solomon code, the minimum distance d satisfies $d = r-k+1$. The condition $r > c^2(k-1)$ is then equivalent to the condition $d > r - r/c^2$. By Theorem 1, C is a c-TA code and traitor tracing amounts to finding a codeword within distance $r - r/c$ of the pirate. Theorem 12 and Corollary 13 of [19] imply that if $t > \sqrt{(k-1)r}$ then all codewords within distance $r - t$ of a given word can be listed in time $O(r^{15})$, and if $t^2 = (1+\delta)(k-1)r$ then the runtime is $O(\frac{r^3}{\delta^6})$. Taking $t = r/c$ gives the desired results. (Note that $k = \log_q N$.)

(ii) The minimum distance d of the code satisfies $d \geq r - k - g + 1$ (see, for example, Theorem 10.6.3 of [28]). By our choice of c we have $d \geq r - k - g + 1 > r - r/c^2$ and $r - r/c < r - \sqrt{r(k+g-1)}$. By Theorem 27 of [19], there exists an algorithm that runs in time polynomial in r that outputs the list of codewords of distance less than $r - \sqrt{r(k+g-1)}$ from a given word. Now apply Theorem 1.

(iii) Theorems 7 and 8 and Corollaries 2 and 3 of [20] imply that there exists an explicit concatenated code over F_q of the correct length r and dimension k, with minimum distance $d \geq (1 - \frac{1}{q})(1-\delta)r$, with a polynomial time list decoding algorithm for e errors, as long as $e < (1 - \sqrt{\delta})(q-1)r/q$. The condition $\delta \leq \frac{q/c^2-1}{q-1}$ implies that $d > r - r/c^2$ and that the upper bound on the number of errors is satisfied when $e \leq r - r/c$. The result now follows from Theorem 1. □

We emphasize that further improvements in the runtime of list decoding algorithms are being rapidly produced. It seems that some of these results will bring the runtime down to $O(r \log^3 r)$ for Reed-Solomon codes, at least in certain cases (see [12]). The list decoding algorithm in [19] for AG codes was improved in [47] (see Theorems 3.4 and 4.1), where an explicit runtime was also given.

4 Comparative Analysis of TA and IPP Traceability

The results in this section justify a focus on TA (as opposed to IPP) schemes. In this paper we have been using the additional structure provided by linear codes to construct schemes for which the TA tracing algorithm is efficient. We know by Lemma 1 that c-TA codes are also c-IPP codes. However the converse fails ([37]; see also Example 1 above). If constructions of schemes for which the IPP tracing algorithm is efficient (i.e., significantly reduced from $O(\binom{N}{c})$ time) are possible, it is reasonable to expect this to be accomplished by introducing an algebraic structure. Here we give evidence that doing so may enable the inherently more efficient TA algorithm to be used to identify traitors. Hence, it is unclear that c-IPP schemes yield any advantage over c-TA schemes in finding a traitor.

First, we prove a necessary condition on Reed-Solomon codes, under which they yield c-TA set systems. This condition is that the minimum distance is greater than $r - r/c^2$, where r is the length of the codewords. This result suggests a potential method for generating examples of schemes that are c-IPP but not c-TA, namely, decreasing the minimum distance. Next we demonstrate through

a family of counterexamples that in fact this approach does not work in general; when the minimum distance is $r - r/c^2$ it is possible to find Reed-Solomon codes for which both the IPP and TA tracing algorithms fail.

We recall that there is a natural way to produce unordered sets from the ordered sets that constitute the code: to a codeword $x = (x_1, \ldots, x_r)$, associate the set $x' = \{(1, x_1), \ldots, (r, x_r)\}$. We define TA and IPP set systems (as opposed to TA and IPP codes) in the natural way, with the noteworthy difference that a pirate *unordered set* consists of r elements such that each element is a member of some coalition member's set. This is a generalization of our earlier definition because it is not necessary to have one element of the form (i, y_i) for each $i = 1, \ldots, r$.

The following theorem is a partial converse of Theorem 1.

Theorem 3. *If $c \geq 2$ is an integer and C is a Reed-Solomon code of length r with minimum distance $d \leq r - \frac{r}{c^2}$, then the set system corresponding to C is not a c-TA set system.*

Proof. As above, if $u \in C$, write $u' = \{(1, u_1), \ldots, (r, u_r)\}$ for the associated element of the set system. Choose a codeword $v = (v_1, \ldots, v_r)$ in C. We will show that a coalition of size at most c exists which does not contain v', but which can implicate v'. In other words, we will construct a pirate set w which can be created by a coalition $\{u'_1, \ldots, u'_b\}$ with $b \leq c$ that does not contain v', but which satisfies $|v' \cap w| \geq |u'_i \cap w|$ for every i. Let $\delta = r - d = k - 1$, where k is the dimension of the code C. By assumption, $\delta \geq r/c^2$.

First, assume $c\delta \leq r$. For $i = 1, \ldots, c$, choose $u_i \in C$, distinct from v, which agrees with v on the δ positions $(i-1)\delta + 1, \ldots, i\delta$. (To do this, simply find a polynomial h_i of degree δ which vanishes on the δ field elements corresponding to these δ positions, and let u_i be the codeword corresponding to the polynomial $f - h_i$, where f is the polynomial corresponding to v.) Notice that, since two distinct codewords can agree on at most δ positions, each u'_i contains at least $r - c\delta$ elements which are not in v' or in u'_j for any $j \neq i$. Since $r - c\delta \geq 0$ and $c \geq 2$, we have $r - c\delta \geq \lceil \frac{r - c\delta}{c} \rceil = \lceil \frac{r}{c} \rceil - \delta$. We can therefore form a pirate set w so that for every i, $|u_i \cap w| \leq \delta + (\lceil \frac{r}{c} \rceil - \delta) = \lceil \frac{r}{c} \rceil$ and $|v' \cap w| = c\delta \geq \lceil \frac{r}{c} \rceil$. Thus the TA algorithm will mark v' as a traitor.

If on the other hand $c\delta > r$, simply choose $u_1, \ldots, u_j$ as above, where $j = \lfloor \frac{r}{\delta} \rfloor < c$, and choose $u_{j+1} \neq v$ to agree with v on the last $r - j\delta$ positions. The coalition $\{u'_1, \ldots, u'_{j+1}\}$ can create v' as a pirate set. □

The previous theorem leaves open the question of whether Reed-Solomon codes with minimum distance at most $r - \frac{r}{c^2}$ might still have traceability when the IPP algorithm is used even though the TA algorithm may no longer correctly identify traitors. The following family of counterexamples illustrates that this is not generally the case. It gives examples of Reed-Solomon codes of length r and minimum distance $r - r/c^2$ which are not c-IPP.

Theorem 4. *Let s and c be positive integers with $c \geq 2$, and let p be a prime number greater than c^2. For $i = 1, \ldots, c$, let $a_i = (i-1)c$. For $i = 1, \ldots, c$, if s*

is not divisible by p, let $g_i(x) = x^s - i$; otherwise let $g_i(x) = x^s + x - i$. Let T be the set of roots of all the c^2 polynomials $g_i - a_j$. Let q be a sufficiently high power of p so that T is a subset of the finite field F_q. Then T consists of $c^2 s$ distinct elements of F_q. Let C be the Reed-Solomon code in which the codewords are the evaluations at the elements of T of all polynomials over F_q of degree at most s. Then the dimension of the code C is $s+1$, the length r of the codewords is $r = c^2 s$, the minimum distance of C is $r - r/c^2$, and C is not c-IPP.

Proof. We first show that T consists of $c^2 s$ distinct elements. Let $h_{ij} = g_i - a_j$. Then $h_{ij}(x) - h_{mn}(x) = -i - (j-1)c + m + (n-1)c$. If $h_{ij}(x) - h_{mn}(x) = 0$, then $m - i$ is divisible by c. Since m and i are both in the range $1, \ldots, c$, they must be equal. Thus $(j-1)c = (n-1)c$, and so $j = n$. Therefore the set $\{h_{ij}\}$ consists of c^2 distinct polynomials of degree s, any two of which differ by a non-zero constant. Therefore no two can have a root in common. Further, the derivative of h_{ij} is sx^{s-1} if s is not divisible by p, and is 1 otherwise. In both cases this derivative is relatively prime to h_{ij} (in the first case, note that h_{ij} is always of the form x^s+(a non-zero constant), so it never has 0 as a root). Therefore all the roots of h_{ij} are simple. So T consists of $c^2 s$ distinct elements, and it makes sense to consider the Reed-Solomon code defined by evaluating polynomials of degree at most s at the elements of T. The code clearly has the stated parameters. The two coalitions corresponding to the polynomials in the sets $\{a_1, \ldots, a_c\}$ and $\{g_1, \ldots, g_c\}$ are disjoint, and each coalition can produce the pirate word defined as follows: for each β in T, the β-th entry of the pirate word is $g_i(\beta) = a_j$, for the unique i and j such that the equality holds. It follows that the code is not c-IPP. □

By evaluating the polynomials at subsets of T of size at least $s+1$ (to ensure that $k \leq r$), we can take the length r to be anything between $s+1$ and $c^2 s$. The resulting minimum distance $r - s$ is then at most $r - r/c^2$.

We remark that if s is not divisible by p, then we can always find a q that works which is a divisor of p^s.

The results in this section lead to the following questions which, while peripheral to the traitor tracing problem, are of independent interest. Is it the case that all Reed-Solomon codes of length r with minimum distance $d \leq r - r/c^2$ are not c-IPP? It is easy to see that this is false for linear codes in general. For example, one-dimensional linear codes are always both c-IPP and c-TA, but can have $d \leq r - r/c^2$ if they are not Reed-Solomon codes (for one-dimensional codes, the minimum distance d is the number of non-zero entries in the non-zero codewords; the codewords of distance less than d from the pirate lie in every coalition that can create the pirate). If the answer to the above question were yes, combining it with Theorem 1 would imply that all Reed-Solomon c-IPP codes are c-TA. We raise as an open question whether all *linear* c-IPP codes are c-TA.

5 Finding All Possible Coalitions

In this section, we describe how a coding theoretic approach can be used to amass additional piracy information: a list of all coalitions that are capable of creating a given pirate. Such information is useful in two respects. It clears all codewords not appearing in any of these coalitions of involvement in constructing the pirate word, and it constitutes useful audit information that may be helpful in the prosecution of a traitor later on. The two algorithms of this section require only that the code have minimum distance greater than $r - \frac{r}{c^2}$, and therefore are applicable to the codes in Theorem 2. The algorithms are fast when fast list decoding techniques exist. In addition, we note that for every code meeting this minimum distance requirement and having fast list decoding, the algorithms enable the IPP traitor tracing algorithm [23,2,37] to run more efficiently (as that algorithm works by intersecting all coalitions that are capable of creating a given pirate word).

At a high level, the first algorithm builds a "tree" from which all c-coalitions capable of constructing a pirate w can be extracted. At the root of the tree lie all codewords that we know must be in *every* such coalition. The children are then candidate codewords for the next member of the coalition. Branches of the tree are extended until the current coalition "covers" w (i.e., is capable of constructing w), or until it becomes clear that this is impossible (e.g., because the coalition is already of size c and still cannot create w). In the latter case that "dead-end" coalition is discarded and other branches of the tree are explored. Before describing the algorithm in more detail, we introduce some of the ideas used. If S is a subset of $\{1, \ldots, r\}$ and $s = |S|$, define a map $f_S : F_q^r \to F_q^{r-s}$ by "forgetting" the entries in positions corresponding to elements of S. If C is a code, then the image code $f_S(C)$ is the *punctured code*, where we view the code C as having been punctured at the positions corresponding to the elements of S. If u is in $f_S(C)$, any codeword v such that $f_S(v) = u$ is called a *lift* of u to C.

We say that U is a *minimal* c-coalition for w if $|U| \leq c$, $w \in \mathsf{desc}(U)$, but w is not in $\mathsf{desc}(V)$ for any proper subset V of U. To obtain all coalitions of size at most c that can create w from the minimal ones, append arbitrary elements of the code.

Algorithm Sketch:
Input: Integer $c > 1$, code C of length r and minimum distance greater than $r - \frac{r}{c^2}$, pirate word $w \in \mathsf{desc}_c(C)$.
Output: A list of coalitions of size at most c that can create w, including all minimal c-coalitions for w.

The basic steps of the algorithm are as follows:

(i) Use list decoding to find all codewords $u_1, \ldots, u_a \in C$ $(a \leq c)$ within distance $r - r/c$ of w. Let S be the subset of $\{1, \ldots, r\}$ on which w agrees with at least one of $\{u_1, \ldots, u_a\}$, and let $s = |S|$. Let $r_1 = r - s$, $c_1 = c - a$, $C_1 = f_S(C)$, and $w_1 = f_S(w)$. (Thus C_1 is the punctured code, r_1 is its length, w_1 is the word which is the image of the pirate word under the

puncturing map, and c_1 is the number of coalition members still to be found.) If $r_1 = 0$, quit and output $\{u_1, \ldots, u_a\}$. Set $i = 1$.

(ii) Use list decoding to find all codewords $v_{i1}, \ldots, v_{ib_i} \in C_i$ ($b_i \leq c_i$) within distance $r_i - r_i/c_i$ of w_i. (Note that the first time this is executed, the output is non-empty.) If this outputs the empty-set, exit to Step (iii). Otherwise, let S_i be the subset of $\{1, \ldots, r_i\}$ on which w_i agrees with v_{ib_i}, and let $s_i = |S_i|$. Let $r_{i+1} = r_i - s_i$, $c_{i+1} = c_i - 1$, $C_{i+1} = f_{S_i}(C_i)$, and $w_{i+1} = f_{S_i}(w_i)$.

(iii) To create the coalitions to output, always start with $u_1, \ldots, u_a$. Then add (a lift to C of) v_{1b_1}, v_{2b_2}, and so on. Continue until the list of codewords "covers" the pirate w. When this process succeeds or dead-ends (i.e., the current list does not yet cover w, but either we cannot find any codewords within the required distance $r_i - r_i/c_i$ of w_i, or we already have c codewords in our list), then move up the "tree" of v_{ij}'s (i.e., move back through the v_{ij}'s) to find the first unexplored branch and continue from there (repeating Step (ii) with a different v_{ij} in place of v_{ib_i}). The algorithm terminates when all branches have been explored.

Analysis of the Algorithm:
The algorithm is correct because the output is clearly a list of coalitions of size at most c that can create the pirate, and includes each minimal c-coalition at least once. (In fact, it may list a coalition more than once.) Note that in Step (iii), all lifts of each v_{ij} should be considered. By Theorem 1, $u_1, \ldots, u_a$ are in every coalition that can create w. In Step (ii), if $d_i > r_i - r_i/c_i^2$ where d_i is the minimum distance of the punctured code C_i, then every coalition that can produce the original pirate w will contain some lift to the original code of some v_{ij}. Moreover, if a lift to C of v_{ij} is in some coalition that can create the original pirate w, then there exists a codeword within $r_i - r_i/c_i$ of v_{ij} (by the pigeonhole principle), and the algorithm will proceed. If Step (ii) returns the empty-set, then the current path is a dead-end. Note that list decoding a punctured code and then lifting accomplishes the same thing as erasure-and-error decoding. When C satisfies any of the sets of conditions in Theorem 2, then Step (i) can be done efficiently (time polynomial in r).

Note that the brute force method for finding all coalitions runs in time $O(crN^c)$, where N is the total number of codewords in the code (for each of the at most N^c coalitions of size at most c, compare each of the r entries of the pirate to the corresponding entry of each member of the coalition). For Reed-Solomon codes with $r = \Theta(c^2 k)$, this gives a runtime of $O(c^3 N^c \log N)$.

Our second algorithm is to list decode to find all codewords $u_1, \ldots, u_a$ ($1 \leq a \leq c$) within distance $r - r/c$ of the pirate (as in Step (i) above), and then use brute force to determine the remaining (at most) $c - a$ members of the coalitions. When C is a Reed-Solomon code satisfying the conditions in Theorem 2(i) with $r = \Theta(c^2 k)$, the dominant term in the runtime is $O(c^3 N^{c-a} \log N)$. This is clearly an improvement over brute force alone, since $a \geq 1$.

6 Future Directions: Tracing with Extra Information

In this section, we describe how other coding theoretic techniques may be applied to the traitor tracing problem when additional information about traitor behavior is available.

One possible approach to tracing traitors is to try to second-guess their strategy. For example, if you believe that one traitor has contributed more than the other members of the coalition to the pirate, you can apply bounded-distance decoding up to the error-correction bound to find such traitors very quickly. This might involve a "ringleader" or "scapegoat" scenario. If on the other hand you believe that all traitors contributed roughly equal amounts, then list decoding should be tried first. Traitors can be searched for in sequences of expanding Hamming balls around the pirate. These searches can be run in parallel or sequentially. The runtime of bounded-distance decoding up to the error-correction bound for Reed-Solomon codes is at most quadratic in the length of the codewords. Note that [32] gives a fast algorithm for list decoding Reed-Solomon codes beyond the error-correction bound (also quadratic in the codeword length), but does not go as far as the Guruswami-Sudan algorithm. It therefore will not be guaranteed to find a traitor, but would quickly find a ringleader.

In [19], list decoding is considered not just in the case of errors, but also in the case of erasures and errors (and another potentially useful case that is referred to as "decoding with uncertain receptions"). For concatenated codes, [20] also deals with the problem of decoding from errors and erasures. Building on [19], [24] presents a high-performance soft-decision list decoding algorithm. We believe that these results also have potential for use in traitor tracing problems, in cases where some additional information is known about the traitors or how they are operating.

If one has information about the traitors or their modes of operation, one can build that information into a reliability matrix, and apply soft-decision decoding algorithms to trace. For example, suppose we know that a traitor who contributed the first entry to the pirate contributed at least r/c entries to the pirate. One can use this information to construct a skewed reliability matrix. If the underlying code is a Reed-Solomon code over a finite field of size q, one can then apply the soft-decision algorithm in [24] to find such a "dominant" traitor. The channel that models this situation is a q-ary symmetric channel. The first column of the reliability matrix will have a 1 in the entry corresponding to the field element that occurs in the first position of the pirate, and 0's elsewhere. For $j > 1$, the jth column of the reliability matrix will have $1 - \epsilon$ in the entry corresponding to the field element in the jth entry of the pirate, and the other entries will all be $\frac{\epsilon}{q-1}$, where $\epsilon < \frac{q-1}{q}$ is chosen so as to optimize the soft-decision decoding algorithm in [24]. If one does not know which entry was contributed by the traitor who contributed the most, one possible search method is to choose entries at random from the pirate and apply the above strategy to search for traitors that contributed that entry.

Erasure-and-error decoding may be useful in fingerprinting or watermarking scenarios, such as those presented in [6,7,15]. In one model, a coalition creates

a pirate copy of the digital content by leaving fixed all codeword entries where they all agree, and choosing the values of the remaining positions from $Q \cup \{?\}$, where Q is the alphabet. The ?'s can be viewed as erasures.

7 Conclusion

We have demonstrated that traitor tracing algorithms can be quite efficient when the construction of the traceability scheme is based on error-correcting codes and the method of tracing is based on fast list decoding algorithms. For the TA algorithm, traitors can be identified in time polynomial in r, where r is roughly $c^2 \log_q N$, rather than in time $O(N)$. In addition, list decoding on successive punctured codes gives a method for identifying all possible traitor coalitions of size at most c more efficiently than a brute force search. This is quite useful because of the additional piracy information it represents, as well as for the efficiency improvements that it enables for another traitor tracing algorithm that has garnered interest recently, the IPP algorithm. We also give evidence for a close relationship between the TA and IPP properties, for linear codes. Finally, we suggest avenues for future research, including explorations of applications of soft-decision and erasure decoding techniques to traitor tracing in scenarios where additional information has been obtained about the traitors or their mode of operation.

Acknowledgments. The authors thank Gui-Leng Feng, Tom Høholdt, Ralf Kötter, and Madhu Sudan for useful conversations.

References

1. N. Alon, J. Bruck, J. Naor, M. Naor and R. Roth. Construction of asymptotically good low-rate error-correcting codes through pseudo-random graphs. *IEEE Transactions on Information Theory* **38** (1992), 509–516.
2. A. Barg, G. Cohen, S. Encheva, G. Kabatiansky and G. Zémor. A hypergraph approach to the identifying parent property: the case of multiple parents, DIMACS Technical Report 2000-20.
3. O. Berkman, M. Parnas and J. Sgall. Efficient dynamic traitor tracing, in 11th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2000), 586–595.
4. E. R. Berlekamp, R. J. McEliece and H. C. A. van Tilborg. On the inherent intractability of certain coding problems. *IEEE Transactions on Information Theory* **24** (1978), 384–386.
5. D. Boneh and M. Franklin. An efficient public key traitor tracing scheme, in "Advances in Cryptology – Crypto '99", *Lecture Notes in Computer Science* **1666** (1999), 338–353.
6. D. Boneh and J. Shaw. Collusion secure fingerprinting for digital data, in "Advances in Cryptology – Crypto '95", *Lecture Notes in Computer Science* **963** (1995), 452–465.
7. D. Boneh and J. Shaw. Collusion secure fingerprinting for digital data, *IEEE Transactions on Information Theory* **44** (1998), 1897–1905.

8. I. Cox, J. Kilian, T. Leighton and T. Shamoon. Secure spread spectrum watermarking for multimedia. *IEEE Transactions on Information Theory* **6** (1997), 1673–1687.
9. B. Chor, A. Fiat and M. Naor. Tracing traitors, in "Advances in Cryptology – Crypto '94", *Lecture Notes in Computer Science* **839** (1994), 480–491.
10. B. Chor, A. Fiat, M. Naor and B. Pinkas. Tracing traitors, *IEEE Transactions on Information Theory* **46** (2000), 893–910.
11. C. Dwork, J. Lotspiech and M. Naor. Digital Signets: Self-Enforcing Protection of Digital Information, in Proc. 28th ACM Symposium on Theory of Computing (STOC 1997), 489–498.
12. G.-L. Feng. Very Fast Algorithms in Sudan Decoding Procedure for Reed-Solomon Codes. Preprint.
13. G.-L. Feng. Fast Algorithms in Sudan Decoding Procedure for Hermitian Codes. Preprint.
14. A. Fiat and M. Naor. Broadcast Encryption, in "Advances in Cryptology – Crypto '93", *Lecture Notes in Computer Science* **773** (1994), 480–491.
15. A. Fiat and T. Tassa. Dynamic traitor tracing, in "Advances in Cryptology – Crypto '99", *Lecture Notes in Computer Science* **1666** (1999), 354–371.
16. E. Gafni, J. Staddon and Y. L. Yin. Efficient methods for integrating traceability and broadcast encryption, in "Advances in Cryptology – Crypto '99", *Lecture Notes in Computer Science* **1666** (1999), 372–387.
17. J. Garay, J. Staddon and A. Wool, Long-Lived Broadcast Encryption, in "Advances in Cryptology – Crypto 2000", *Lecture Notes in Computer Science* **1880** (2000), 333-352.
18. V. D. Goppa. Geometry and codes. Kluwer Academic Publishers, Dordrecht, 1988.
19. V. Guruswami and M. Sudan. Improved decoding of Reed-Solomon and algebraic-geometry codes, *IEEE Transactions on Information Theory* **45**(6) (1999), 1757–1767.
20. V. Guruswami and M. Sudan. List decoding algorithms for certain concatenated codes, in Proc. 32nd ACM Symposium on Theory of Computing (STOC 2000), 181–190.
21. T. Høholdt and R. R. Nielsen. Decoding Reed-Solomon codes beyond half the minimum distance, in Coding theory, cryptography and related areas (Guanajuato, 1998), Springer, Berlin (2000), 221–236.
22. T. Høholdt and R. R. Nielsen. Decoding Hermitian codes with Sudan's algorithm. To appear in the 13th AAECC Symposium.
23. H. D. L. Hollmann, J. H. van Lint, J-P. Linnartz and L. M. G. M. Tolhuizen. On codes with the identifiable parent property, *Journal of Combinatorial Theory A* **82** (1998), 121–133.
24. R. Koetter and A. Vardy. Algebraic soft-decision decoding of Reed-Solomon codes. Preprint.
`http://www.dia.unisa.it/isit2000/lavori/455.ps`.
25. R. Kumar, S. Rajagopalan and A. Sahai. Coding constructions for blacklisting problems without computational assumptions, in "Advances in Cryptology – Crypto '99", *Lecture Notes in Computer Science* **1666** (1999), 609–623.
26. K. Kurosawa, M. Burmester and Y. Desmedt. A proven secure tracing algorithm for the optimal KD traitor tracing scheme. DIMACS Workshop on Management of Digital Intellectual Properties, April, 2000, and Eurocrypt 2000 rump session.
27. K. Kurosawa and Y. Desmedt. Optimal traitor tracing and asymmetric schemes, in "Advances in Cryptology – Eurocrypt '98", *Lecture Notes in Computer Science* **1438** (1998), 145–157.

28. J. H. van Lint. Introduction to coding theory. Third edition. Graduate Texts in Mathematics **86**, Springer-Verlag, Berlin (1999).
29. M. Naor and B. Pinkas. Efficient Trace and Revoke Schemes, to appear in Proceedings of Financial Cryptography 2000.
30. V. Olshevsky and A. Shokrollahi. A displacement structure approach to efficient decoding of algebraic geometric codes, in Proc. 31st ACM Symposium on Theory of Computing (STOC 1999), 235–244.
31. B. Pfitzmann. Trials of traced traitors, in Information Hiding, First International Workshop, *Lecture Notes in Computer Science* **1174** (1996), 49–64.
32. R. M. Roth and G. Ruckenstein. Efficient decoding of Reed-Solomon codes beyond half the minimum distance. *IEEE Transactions on Information Theory* **46** (2000), 246–257.
33. R. Safavi-Naini and Y. Wang. Sequential Traitor Tracing, in "Advances in Cryptology – CRYPTO 2000", *Lecture Notes in Computer Science* **1880** (2000), 316–332.
34. B.-Z. Shen. A Justesen construction of binary concatenated codes that asymptotically meet the Zyablov bound for low rate, *IEEE Transactions on Information Theory* **39** (1993), 239–242.
35. M. A. Shokrollahi and H. Wassermann. Decoding Algebraic-Geometric Codes Beyond the Error-Correction Bound, in Proc. 30th ACM Symposium on Theory of Computing (STOC 1998), 241–248.
36. M. A. Shokrollahi and H. Wassermann. List Decoding of Algebraic-Geometric Codes. *IEEE Transactions on Information Theory* **45** (1999), 893–910.
37. J. N. Staddon, D. R. Stinson and R. Wei. *Combinatorial properties of frameproof and traceability codes*. To appear in *IEEE Transactions on Information Theory*.
38. H. Stichtenoth. Algebraic Function Fields and Codes. Springer-Verlag, Berlin, 1993.
39. D. R. Stinson, Tran van Trung and R. Wei. Secure frameproof codes, key distribution patterns, group testing algorithms and related structures, *Journal of Statistical Planning and Inference* **86** (2000), 595-617.
40. D. R. Stinson and R. Wei. Combinatorial properties and constructions of traceability schemes and frameproof codes, *SIAM Journal on Discrete Mathematics* **11** (1998), 41–53.
41. D. R. Stinson and R. Wei. Key preassigned traceability schemes for broadcast encryption, in "Selected Areas in Cryptology – SAC '98", *Lecture Notes in Computer Science* **1556** (1999), 144–156.
42. M. Sudan. Decoding of Reed Solomon codes beyond the error-correction bound. *Journal of Complexity* **13**(1) (1997), 180–193.
43. M. Sudan. Decoding of Reed Solomon codes beyond the error-correction diameter, in Proc. 35th Annual Allerton Conference on Communication, Control and Computing (1997), 215–224.
44. M. A. Tsfasman and S. G. Vlăduţ. Algebraic-geometric codes. Kluwer Academic Publishers, Dordrecht, 1991.
45. M. A. Tsfasman, S. G. Vlăduţ and Th. Zink. Modular curves, Shimura curves, and Goppa codes, better than Varshamov-Gilbert bound. *Math. Nachr.* **109** (1982), 21–28.
46. W.-G. Tzeng and Z.-J. Tzeng. A Traitor Tracing Scheme Using Dynamic Shares, to appear in PKC2001.
47. X.-W. Wu and P. H. Siegel. Efficient List Decoding of Algebraic Geometric Codes Beyond the Error Correction Bound, submitted to *IEEE Transactions on Information Theory*.
48. F. Zane. Efficient Watermark Detection and Collusion Security, to appear in Proceedings of Financial Cryptography 2000.