On Algorithms for Searching a Consistent Set of Shares in a Threshold Scheme and the Related Covering Problem

Raylin Tso[†], Ying Miao[‡] and Eiji Okamoto[††]

E-mail: [†] raylin@cipher.risk.tsukuba.ac.jp
[‡] miao@risk.tsukuba.ac.jp
[††] okamoto@risk.tsukuba.ac.jp

Graduate School of Systems and Information Engineering
University of Tsukuba
Tsukuba 305-8573 Japan

Abstract

In the Shamir (k, n) threshold scheme, if one or more of the n shares are fake, then the secret may not be reconstructed correctly by some sets of k shares. Supposing that at most t of the n shares are fake, Rees et al. (1999) described two algorithms to determine consistent sets of shares so that the secret can be reconstructed correctly from k shares in any of these consistent sets. In their algorithms, no honest participant can be absent and at least $n - t$ shares should be pooled during the secret reconstruction phase. In this paper, we propose a modified algorithm for this problem so that the number of participants taking part in the secret reconstruction can be reduced to $k + 2t$ and the shares need to be pooled can be reduced to, in the best case, $k + t$, and less than or equal to $k + 2t$ in the others. Its performance is evaluated. A construction for t-coverings, which play key roles in these algorithms, is also provided.

Key words: algorithm; consistent set; covering; threshold scheme.

1 Introduction

Secret sharing schemes are indispensable whenever secret information needs to be kept collectively by a group of participants in such a way that only authorized subgroups of participants are able to reconstruct the secret. Typical examples of such schemes are threshold schemes introduced by Blakley [1] and Shamir [15] independently in 1979. Informally, a (k, n) *threshold scheme* is a method of breaking a secret K up into n different shares $S_1, S_2, \ldots, S_n$ such that

1. with the knowledge of any k or more shares ($k \leq n$), the secret K can be easily derived; and

2. with the knowledge of any $k-1$ or fewer shares, it is impossible to derive the secret K.

The Shamir threshold scheme is based on the Lagrange interpolating polynomial where the constant term is the secret K. In this paper we will adopt the Shamir scheme for the convenience of description.

The problem of detecting cheaters in threshold schemes was investigated by McEliece and Sarwate [11], and many others (see, for example, [2], [5], [12], [17]). Under the assumption that at most t of the n shares are fake, by recognizing that the secret and the n shares altogether form a codeword in an $[n+1, k, n+2-k]$ maximum distance separable code, McEliece and Sarwate [11] showed that if $k+2t$ shares are pooled together, then there exists a unique polynomial of degree at most $k-1$ so that at least $k+t$ of the genuine shares lie on it. This unique polynomial, which is used to compute the secret, can be efficiently determined by both the Berlekamp-Massey algorithm or the Euclidean algorithm for Reed-Solomon codes.

Rees et al. [13] considered the problem of determining consistent sets of shares in a threshold scheme with cheaters. A set of shares in a (k, n) threshold scheme is called *consistent* if every k-subset of the shares derives the same secret. Rees et al. proposed a deterministic algorithm, based on a particular class of t-coverings with index 1, to reconstruct the secret of the Shamir (k, n) threshold scheme when some shares are fake. They also considered a randomized algorithm for the same problem, and compared it with the above mentioned deterministic algorithm. Their underlying idea is to find a suitable set system $(S, \mathcal{T})$, where S is the set of all n shares and $\mathcal{T}$ is a collection of k-subsets of S, so that for any t-subset S_t of shares (thus the subset of all fake shares, if we assume that at most t of the n shares are fake), there is at least one $T \in \mathcal{T}$ such that T does not contain any share in this t-subset S_t. Then the k-subset of $\mathcal{T}$ containing no fake shares can be used to derive the secret correctly.

Unfortunately, one drawback of Rees et al. 's algorithms is that they sacrifice the property of *threshold*. That is, no honest participant can be absent if they decide to reconstruct the secret, while in a conventional (k, n) threshold scheme, only k of the n participants are needed to pool their shares. As a consequence, Rees et al. 's algorithms are not so practical in a (k, n) threshold scheme where k and t are small but n is relatively large, or gathering all the participants together is impossible.

In this paper, we propose a modified algorithm for this problem so that the number of participants taking part in the secret reconstruction can be reduced to $k+2t$ if at most t of the n shares are fake. In our new algorithm, the shares need to be pooled can be reduced to, in the best case, $k+t$, and less than or equal to $k+2t$ in the others. Some efficiency will be sacrificed in our algorithm in order to allow any $k+2t$ of the n participants to achieve the end of determining a consistent set of shares in a threshold scheme with at most t cheaters. This is verified by the numerical results of our computer experiments. The expected number of shares used to reconstruct the secret will also be compared with those of Rees et al.

's algorithms [13] and $k + 2t$, the number of shares needed to reconstruct the secret by McEliece and Sarwate's method [11], in some special cases. Finally, a construction for those set systems used in the algorithms mentioned above is also presented.

Since many notations and symbols are used in this paper, a list of them is given in Appendix 2.

2 Rees et al. 's two algorithms

Suppose that the Shamir (k, n) threshold scheme is implemented in $GF(q)$. Let

$$S = \{(x_i, y_i) : \ 1 \le i \le n\} \subseteq (GF(q) \setminus \{0\}) \times GF(q)$$

be the set of n shares, and assume that at most t of the n shares are fake. That is, there exists a polynomial $P_0(x) \in GF(q)[x]$ of degree at most $k-1$ such that $y_i = P_0(x_i)$ for at least $n - t$ of the n shares. The secret, which can be reconstructed from any k genuine shares, is the value $P_0(0)$.

Denote the subset of *good shares* by $G = \{i : \ y_i = P_0(x_i), 1 \le i \le n\}$, and the subset of *bad shares* by $B = \{1, 2, \ldots, n\} \setminus G$. Then $|G| \ge n - t$ and $|B| \le t$.

For any $T \subseteq \{1, 2, \ldots, n\}$ such that $|T| = k$, there is a unique polynomial P_T of degree at most $k-1$ such that $P_T(x_i) = y_i$ for all $i \in T$, which can easily be computed by Lagrange interpolation

$$P_T(x) = \sum_{i \in T} y_i \prod_{j \in T \setminus \{i\}} \frac{x - x_j}{x_i - x_j}.$$

The following two facts are obvious:

1. If $T \subseteq G$, then $P_T = P_0$.
2. If $T \cap B \ne \emptyset$, then $P_T \ne P_0$.

Define $C_T = \{i : \ P_T(x_i) = y_i, 1 \le i \le n\}$. Then clearly $|C_T| \ge n - t$ if $T \subseteq G$, and $|C_T| \le k + t - 1$ if $T \cap B \ne \emptyset$. If $n - t \le k + t - 1$, then there could exist a polynomial $P_T \ne P_0$ of degree at most $k-1$ such that at least $n - t$ shares lie on P_T. Therefore, Rees et al. [13] required that the inequality $n \ge k + 2t$ always holds.

Let v, k, λ and t be positive integers such that $v \ge k \ge t$. A t-(v, k, λ) *covering* is a pair $(\mathcal{V}, \mathcal{B})$, where $\mathcal{V}$ is a v-set of elements, called *points*, and $\mathcal{B}$ is a collection of k-subsets of $\mathcal{V}$, called *blocks*, such that every t-subset of points occurs in at least λ blocks of $\mathcal{B}$. The *covering number* $C_\lambda(v, k, t)$ is the minimum number of blocks in any t-(v, k, λ) covering. A t-(v, k, λ) covering $(\mathcal{V}, \mathcal{B})$ is *optimal* if $|\mathcal{B}| = C_\lambda(v, k, t)$.

Let $\mathcal{T}$ be a collection of k-subsets of $\{1, 2, \ldots, n\}$ such that its complement $\{\{1, 2, \ldots, n\} \setminus T : \ T \in \mathcal{T}\}$ forms the collection of blocks of a t-$(n, n-k, 1)$ covering. Then Rees et al. [13] claimed that the following deterministic algorithm can compute the polynomial $P_0(x) \in GF(q)[x]$.

Algorithm 1
Input T, S, n, k, t.
For each $T \in \mathcal{T}$, perform the following steps:

1. compute P_T
2. compute C_T
3. **if** $|C_T| \geq n - t$ **then** $P_0 = P_T$ and QUIT

Rees et al. [13] also provided a randomized algorithm to compute $P_0(x) \in GF(q)[x]$.

Algorithm 2
Input S, n, k, t.
REPEAT the following steps:

1. let T be a random k-subset of $\{1, 2, \ldots, n\}$
2. compute P_T
3. compute C_T
4. **if** $|C_T| \geq n - t$ **then**
   $P_0 = P_T$ and QUIT
   else
   proceed to the next iteration of the REPEAT loop

Algorithm 2 is not as efficient as Algorithm 1 because $C_1(n, n-k, t)$ provides an upper bound on the number of iterations required by Algorithm 1, whereas Algorithm 2 is a Las Vegas algorithm which may not give an answer.

3 The new practical algorithm

As mentioned in the first section, a drawback of Rees et al. 's algorithms is that all the honest participants should be gathered together in order to make the secret reconstruction successful in any situation. On the other hand, we also note that if we apply Algorithm 1 of Rees et al. 's method to a (k, n_s) multiple threshold scheme of s multiplicity with at most t cheaters, or apply to s different (k, n_s) threshold schemes with at most t cheaters each and the same threshold value k, but varying in the number of participants n_s, then we need to construct and store s different (optimal) t-$(n_s, n_s - k, 1)$ coverings at first. But some of the required t-$(n_s, n_s - k, 1)$ coverings may be unknown or hard to be constructed. In addition, storing all the data of these coverings may consume a lot of capacity or cost expensively. In this section, we improve Rees et al. 's Algorithm 1 so that the new algorithm can overcome all the disadvantages described above.

According to Rees et al. [13], as was described in Section 2, $|C_T| \leq k + t - 1$ if $T \cap B \neq \emptyset$. Therefore, if $|C_T| \geq k + t$, then it must be the case that $T \cap B = \emptyset$. On the other hand, if we define $NC_T = \{i : P_T(x_i) \neq y_i, 1 \leq i \leq n\}$, then $|NC_T| \leq t$ if $T \cap B = \emptyset$. As a consequence, the following two facts become obvious:

1. If $|C_T| \geq k+t$, then $P_T = P_0$.

2. If $|NC_T| \geq t+1$, then $P_T \neq P_0$.

Let $\mathcal{R}$ be a $(k+2t)$-subset of $\{1,2,\ldots,n\}$, $S_{\mathcal{R}} = \{(x_i, y_i) : i \in \mathcal{R}\} \subseteq S$ and $\mathcal{T}$ be a collection of k-subsets of $\mathcal{R}$ such that its complement $\{\mathcal{R} \setminus T : T \in \mathcal{T}\}$ is the collection of blocks of a t-$(k+2t, 2t, 1)$ covering. Then the following deterministic algorithm will compute the polynomial $P_0(x)$ for any $k+2t$ of the n participants. In this algorithm, for convenience, we denote $\mathcal{R} \setminus T = \{r_{i_1}, \ldots, r_{i_{2t}}\}$ for each $T \in \mathcal{T}$, where the subscripts are ordered randomly.

Algorithm 3
Input $\mathcal{R}, \mathcal{T}, S_{\mathcal{R}}, k, t$.
For each $T \in \mathcal{T}$, perform the following steps:

1. compute P_T

2. $|NC_T| = 0$

3. $|C_T| = k$

4. **for** $j = 1$ **to** $2t$ **do**
   if $y_{r_{i_j}} = P_T(x_{r_{i_j}})$, **then** $|C_T| + +$
   else $|NC_T| + +$
   if $|C_T| \geq k+t$, **then** $P_0 = P_T$ and QUIT
   else if $|NC_T| \geq t+1$ **then** BREAK

The advantage of adding the computation of NC_T is that if $P_T(x) \neq P_0(x)$, then the *for* loop can be broken out at the point $|NC_T| = t+1$ and the computation for the next T can be proceeded, while in Algorithm 1 and Algorithm 2, $P_T(x) \neq P_0(x)$ will lead the algorithms to run without stop until all the shares have been pooled and result in $|C_T| < k+t$. Another advantage is that one and only one of the two conditions $|C_T| \geq k+t$ and $|NC_T| \geq t+1$ is satisfied in each loop, and there is at least one k-subset $T \in \mathcal{T}$ such that $|C_T| \geq k+t$, so adding the computation of NC_T makes any $k+2t$ of the n participants possible to reconstruct the secret.

For each of the three algorithms, the best case occurs when the secret is successfully reconstructed at the first try and the algorithm is ended in $P_T(x) = P_0(x)$ with all the pooled shares belonging to C_T. In this case, only $k+t$ shares are needed in Algorithm 3 while $n-t$ shares are needed in Algorithm 1 and Algorithm 2.

4 Performance evaluation of the three algorithms

In this section, we assume that there are exactly t bad shares. We evaluate these three algorithms for their efficiencies and the total shares need to be pooled.

Let β_r be the expected number of iterations of the randomized Algorithm 2. Then it is not difficult to know (see [13]) that

$$\beta_r = \frac{\binom{n}{k}}{\binom{n-t}{k}}.$$

However, the expected numbers of iterations of Algorithm 1 and Algorithm 3 depend on the specific set systems $\mathcal{T}$ they used. Since there is no uniform description of "good" t-$(n, k, 1)$ coverings, we can not expect to find a unified theoretical method for the computation of the expected numbers of iterations of Algorithm 1 and Algorithm 3. Instead, we set up a computer experiment to compare the expected number β_{d_3} of iterations of the deterministic Algorithm 3 with β_{d_1} and β_r, those of Algorithms 1 and 2, and to compare the average number s_{d_3} of shares needed in Algorithm 3 to reconstruct the secret correctly with s_{d_1}, that needed in Algorithm 1, and the value $k + 2t$, the number of shares needed in McEliece and Sarwate's method [11].

To avoid any source of bias, in our experiment, we neither construct any specific polynomial $P_0(x) \in GF(p)[x]$ nor construct all the genuine shares (x_i, y_i) for $i \in G$ and all the fake shares (x_j, y_j) for $j \in B$. This causes no problem for the computation of the expected number of iterations of each algorithm. But when we compute the expected number of shares needed in Algorithm 3, for some k-subset T of $\mathcal{R}$ with $P_T(x) \neq P_0(x)$, we will not be able to verify whether a share belongs to C_T or NC_T without the knowledge of the polynomial $P_T(x)$ and all the shares belonging to $S_{\mathcal{R}}$. Therefore, we make an assumption that if $P_T(x) \neq P_0(x)$, then in Algorithm 3, the good shares not belonging to T will be counted in NC_T. This assumption is reasonable if the Shamir threshold scheme is implemented in $GF(q)$ with $q >> max\{k, 2t\}$. This will be explained right after Lemma 4.1. We also assume that the bad shares not belonging to T will be counted in C_T in Algorithm 3 if $P_T(x) \neq P_0(x)$, since any cheater wants to mislead other participants to some incorrect secret and thus he/she will try his/her best to make his/her bad share belong to C_T, which, as a by-product, forces more shares to be pooled. This assumption also results in an overestimation of the number of shares needed in Algorithm 3 if the fake shares are caused by communication noise instead of the existence of cheaters.

Lemma 4.1 Let $P_0(x) \in GF(q)[x]$ and $P_T(x) \in GF(q)[x]$ be two different polynomials of degree at most $k-1$. For any randomly chosen u-subset of points from $P_0(x) \in GF(q)[x]$, if $q >> max\{k, u\}$, then the probability that at least one of these u points lies on $P_T(x)$ is nearly equal to 0.

Proof The number of points which lie on both $P_0(x)$ and $P_T(x)$ is at most $k-1$ and there are totally q points on $P_0(x) \in GF(q)[x]$ and on $P_T(x) \in GF(q)[x]$, respectively. Let Q_0 and Q_T denote the sets of these q points respectively. Then, for any randomly chosen u-subset $U \subseteq Q_0$, the probability that $U \cap Q_T \neq \emptyset$ will be

$$Prob(U \cap Q_T \neq \emptyset) = 1 - Prob(U \cap Q_T = \emptyset)$$

$$\leq 1 - \frac{\binom{q-k+1}{u}}{\binom{q}{u}} = 1 - \frac{(q-k+1)\times\cdots\times(q-k+2-u)}{q\times\cdots\times(q-u+1)}$$
$$\approx 0,$$

where the inequality comes from $|Q_0 \cap Q_T| \leq k-1$. □

In our experiment for Algorithm 3, $|(\mathcal{R}\setminus T)\cap G| \leq |\mathcal{R}\setminus T| = 2t$ holds for any $T \in \mathcal{T}$, and if $|T\cap B| = k_1 > 0$, then $|T\cap G| = k - k_1$. Meanwhile, $|\{(x_i, y_i) : y_i = P_0(x_i), x_i \neq 0\}| = q-1$, and at most $k-1$ of these $q-1$ points may lie on $P_T(x)$ because the degrees of $P_0(x)$ and $P_T(x)$ are at most $k-1$ and $P_0(x) \neq P_T(x)$. Since $k-k_1$ of these common points have been pooled, by Lemma 4.1, the probability that at least one of the remaining (at most k_1-1) good shares lying on $P_T(x)$ can be included in the $2t$-set $\mathcal{R}\setminus T$ is nearly equal to 0, and therefore it can be ignored.

Table 1 reports the numerical results of our experiment. The values of β_{d_1} and β_r are taken from Table 4 of Rees et al. [13]. The twenty-seven t-$(n, n-k, 1)$ coverings with different parameters n and k used in Algorithm 1 are listed below:
$k = 3$ and $t = 2$: the following blocks form a 2-$(n, n-3, 1)$ covering over $\{1, 2, \ldots, n\}$ for any $n \geq 9$:

$$\{1,2,\ldots,n\}\setminus\{1,2,3\},\quad \{1,2,\ldots,n\}\setminus\{4,5,6\},\quad \{1,2,\ldots,n\}\setminus\{7,8,9\}.$$

$k = 4$ and $t = 2$: the following blocks form a 2-$(n, n-4, 1)$ covering over $\{1, 2, \ldots, n\}$ for any $n \geq 12$:

$$\{1,2,\ldots,n\}\setminus\{1,2,3,4\},\quad \{1,2,\ldots,n\}\setminus\{5,6,7,8\},$$
$$\{1,2,\ldots,n\}\setminus\{9,10,11,12\}.$$

$k = 4$ and $t = 3$: the following blocks form a 3-$(n, n-4, 1)$ covering over $\{1, 2, \ldots, n\}$ for any $n \geq 16$:

$$\{1,2,\ldots,n\}\setminus\{1,2,3,4\},\quad \{1,2,\ldots,n\}\setminus\{5,6,7,8\},$$
$$\{1,2,\ldots,n\}\setminus\{9,10,11,12\},\quad \{1,2,\ldots,n\}\setminus\{13,14,15,16\}.$$

For the set systems used in Algorithm 3, we use three coverings from the web page [18]: a 2-$(7, 4, 1)$ covering with $k = 3$ and $t = 2$, a 2-$(8, 4, 1)$ covering with $k = 4$ and $t = 2$, and a 3-$(10, 6, 1)$ covering with $k = 4$ and $t = 3$, which are listed below.
$\mathcal{R} = \{1, 2, 3, 4, 5, 6, 7\}$: the following blocks form a 2-$(7, 4, 1)$ covering over $\mathcal{R}$:

$$\{1,2,3,4\},\quad \{1,4,5,6\},\quad \{1,5,6,7\},$$
$$\{2,3,4,7\},\quad \{2,3,5,6\}.$$

Table 1:

No	n	k	t	β_{d_1}	β_r	β_{d_3}	s_{d_1}	s_{d_3}	$k+2t$
1	9	3	2	1.833	2.400	2.145	8.832	6.443	7
2	10	3	2	1.733	2.143	2.041	9.813	6.428	7
3	11	3	2	1.655	1.964	1.926	10.836	6.227	7
4	12	3	2	1.591	1.833	1.800	11.863	6.154	7
5	13	3	2	1.538	1.733	1.701	12.871	6.065	7
6	59	3	2	1.105	1.111	1.134	58.957	5.250	7
7	109	3	2	1.056	1.058	1.068	108.981	5.132	7
8	159	3	2	1.038	1.039	1.045	158.989	5.097	7
9	209	3	2	1.029	1.029	1.031	208.988	5.053	7
10	12	4	2	1.818	2.357	2.277	11.885	7.336	8
11	13	4	2	1.744	2.167	2.258	12.895	7.294	8
12	14	4	2	1.681	2.022	2.132	13.892	7.216	8
13	15	4	2	1.629	1.909	1.948	14.906	7.121	8
14	16	4	2	1.583	1.818	1.921	15.902	7.085	8
15	62	4	2	1.134	1.144	1.192	61.963	6.287	8
16	112	4	2	1.073	1.076	1.104	111.984	6.160	8
17	162	4	2	1.050	1.051	1.082	161.983	6.126	8
18	262	4	2	1.031	1.031	1.041	261.991	6.064	8
19	16	4	3	2.036	2.545	2.582	15.880	9.132	10
20	17	4	3	1.956	2.378	2.467	16.884	8.985	10
21	18	4	3	1.887	2.242	2.393	17.864	8.944	10
22	19	4	3	1.828	2.130	2.291	18.867	8.886	10
23	20	4	3	1.775	2.036	2.171	19.899	8.813	10
24	66	4	3	1.196	1.210	1.274	65.956	7.592	10
25	116	4	3	1.108	1.112	1.135	115.990	7.329	10
26	216	4	3	1.057	1.058	1.079	215.986	7.179	10
27	316	4	3	1.039	1.039	1.056	315.984	7.139	10

$\mathcal{R} = \{1,2,3,4,5,6,7,8\}$: the following blocks form a 2-$(8,4,1)$ covering over $\mathcal{R}$:

$$\{1,2,3,4\},\ \{1,4,5,8\},\ \{1,5,6,7\},$$
$$\{2,3,5,8\},\ \{2,3,6,7\},\ \{4,6,7,8\}.$$

$\mathcal{R} = \{1,2,3,4,5,6,7,8,9,10\}$: the following blocks form a 3-$(10,6,1)$ covering over $\mathcal{R}$:

$$\{1,2,3,4,5,7\},\ \{1,2,3,5,9,10\},\ \{1,2,3,4,6,10\},$$
$$\{1,2,4,8,9,10\},\ \{1,3,7,8,9,10\},\ \{1,5,6,7,8,9\},$$
$$\{2,3,4,5,6,8\},\ \{2,6,7,8,9,10\},\ \{3,4,5,6,7,9\},$$
$$\{4,5,6,7,8,10\}.$$

In the experiment, the bad shares are selected randomly in each of the cases. $T \in \mathcal{T}$ in Algorithm 1 and Algorithm 3, and the $(k+2t)$-subset $\mathcal{R}$ of the n shares used in Algorithm 3, are also chosen randomly.

It turns out that the efficiencies of the three algorithms are very close. Although Algorithm 1 is usually a little more efficient than Algorithm 3, the expected number of shares needed in Algorithm 1 is much larger than that needed in Algorithm 3, especially for large n. Also note that s_{d_3} is smaller than $k+2t$, the number of shares needed in McEliece and Sarwate [11].

5 Constructions for optimal or nearly optimal coverings

In Algorithms 1 and 3, t-(v,k,λ) coverings with $\lambda = 1$ are required. Obviously, any t-(v,k,λ) covering is also a t-$(v,k,1)$ covering. But from Appendix 1, we can know that Algorithm 1 would succeed efficiently only when the numbers of blocks containing t-subsets of points are distributed as evenly as possible. Similar statements should also hold for Algorithm 3. If the number of blocks in a t-$(v,k,1)$ covering is b, then it is desirable that the number of blocks containing a t-subset is $\lfloor b\binom{k}{t}/\binom{v}{t} \rfloor$ or $\lfloor b\binom{k}{t}/\binom{v}{t} \rfloor + 1$, where $\lfloor x \rfloor$ denotes the greatest integer smaller than or equal to x, i.e., we wish the t-$(v,k,1)$ covering to be a t-$(v,k,\lfloor b\binom{k}{t}/\binom{v}{t} \rfloor)$ covering. However, it may happen that the existence of an optimal or a nearly optimal t-$(v,k,1)$ covering is unknown, but that of an optimal or a nearly optimal t-$(v,k,\lfloor b\binom{k}{t}/\binom{v}{t} \rfloor)$ covering has been established already, or is easy to be established, although they have the same parameters v, k and t. For example, the existence of an optimal 2-$(v,5,1)$ covering with $v \equiv 0 \pmod 4$ is not settled in general, but the existence of an optimal 2-$(v,5,2)$ covering and the existence of an optimal 2-$(v,5,3)$ covering, all with $v \equiv 0 \pmod 4$, had been settled for a while (see, for example, [10, 16]). A well-known fact is that only finitely many t-$(v,k,1)$ designs with $t \geq 3$ are known and none is known for $t \geq 6$ (see [3]),

while infinitely many t-(v,k,λ) designs with $t \geq 3$ and $\lambda > 1$ are known (see [7]). Here a t-(v,k,λ) design means a t-(v,k,λ) covering where any t-subset of points is contained in exactly λ blocks. Clearly, constructing optimal or nearly optimal t-(v,k,λ) coverings with $t \geq 3$ and $\lambda > 1$ is much more easier than constructing optimal or nearly optimal t-$(v,k,1)$ coverings with $t \geq 3$.

In order for Algorithms 1 and 3 to succeed more efficiently, we try to construct optimal or nearly optimal t-$(n,n-k,\lambda)$ coverings with $2t \leq n-k \leq kt$. The first inequality was required for the uniqueness of the polynomial of degree at most $k-1$ such that at least $n-t$ shares lie on it. The second one is because that $C_1(n,n-k,t) = t+1$ for $n-k \geq kt$ had been already proved by Mills [9], so that the applications of Algorithms 1 and 3 are sufficient for efficiently searching consistent sets of shares. In what follows, we describe a construction for optimal or nearly optimal 3-$(n,n-k,\lambda)$ coverings with $6 \leq n-k \leq 3k$ by inflating the blocks of the ingredient coverings. It is a variant of Ling's union construction [8] for t-designs. Such a construction method had been used years ago by Kageyama and Miao [6] for resolvable 2-designs.

We recap some concepts in combinatorial design theory. A *resolution class* in a t-(v,k,λ) design $(\mathcal{V},\mathcal{B})$ is a sub-collection of blocks in $\mathcal{B}$ that partition the point set $\mathcal{V}$. A *resolvable* t-(v,k,λ) design is a t-(v,k,λ) design whose collection of blocks can be partitioned into resolution classes.

Theorem 5.1 *Let* $\lambda' = \mu\lceil\lambda\frac{(n-1)(n-2)}{(k-1)(k-2)}\rceil + 3\mu\frac{l(n-1)}{l-2}\lceil\lambda\frac{n-2}{k-2}\rceil + \lambda\mu(\frac{(nl-1)(nl-2)}{(l-1)(l-2)} - 1 - 3\frac{l(n-1)}{l-2})$, *where* $\lceil x\rceil$ *denotes the smallest integer greater than or equal to* x. *If there exists a resolvable* 3-(nl,l,μ) *design, then*

$$C_{\lambda'}(nl,kl,3) \leq \mu\frac{(nl-1)(nl-2)}{(l-1)(l-2)}C_\lambda(n,k,3).$$

Proof Let $(\mathcal{V},\mathcal{B})$ be a resolvable 3-(nl,l,μ) design with resolution classes $R_1, R_2, \ldots, R_{\mu\frac{(nl-1)(nl-2)}{(l-1)(l-2)}}$, where $R_i = \{B_{i1}, B_{i2}, \ldots, B_{in}\}$ for $i = 1,2,\ldots,\mu\frac{(nl-1)(nl-2)}{(l-1)(l-2)}$. On $R_i = \{B_{i1}, B_{i2}, \ldots, B_{in}\}$, we construct an optimal 3-(n,k,λ) covering, where each B_{ij} is viewed as a point. The resultant block corresponding to the block B in the optimal 3-(n,k,λ) covering is $\cup_{B_{ij}\in B}B_{ij}$, where the union is taken over $\mathcal{V}$. Then it can be checked that the resultant set system is the desired 3-(nl,kl,λ') covering, with a total of $\mu\frac{(nl-1)(nl-2)}{(l-1)(l-2)}C_\lambda(n,k,3)$ blocks. We are to check that every 3-subset of points is contained in at least λ' blocks. Let $\{x_1,x_2,x_3\}$ be an arbitrary 3-subset of $\mathcal{V}$. The number of resolution classes in the resolvable 3-(nl,l,μ) design is $\mu\frac{(nl-1)(nl-2)}{(l-1)(l-2)}$, and μ of the resolution classes contain $\{x_1,x_2,x_3\}$ as a subset in a block of the resolution class. Hence, the 3-subset $\{x_1,x_2,x_3\}$ appears at least $\lceil\lambda\frac{(n-1)(n-2)}{(k-1)(k-2)}\rceil$ times in the blocks of the optimal 3-(n,k,λ) covering defined on each of these μ resolution classes. There are $\mu\frac{nl-2}{l-2} - \mu$ resolution classes such that $\{x_1,x_2\}$ appears in a block but $\{x_3\}$ does not appear in that block. In the optimal 3-(n,k,λ) covering defined on each of these resolution classes, the 3-subset $\{x_1,x_2,x_3\}$ appears at least in $\lceil\lambda\frac{n-2}{k-2}\rceil$ blocks. For each of the remaining resolution classes, the 3-subset $\{x_1,x_2,x_3\}$ appears at least λ times in the blocks

of the optimal 3-(n,k,λ) covering defined on it. Totally, the 3-subset $\{x_1,x_2,x_3\}$ appears at least in λ' resultant blocks. □

The existence of resolvable 3-$(4n,4,1)$ designs is summarized in [3].

Theorem 5.2 *A resolvable* 3-$(4n,4,1)$ *design exists for every* $n \equiv 1,2 \pmod 3$ *except possibly for* $n \in P = \{55,59,73,91,115,149,169,181,269,275,313,329,455,559,577,581,595,635,685,703,905,955,1589\}$.

Applying Theorem 5.1 with Theorem 5.2, we obtain the following result.

Corollary 5.3 *Let* $\lambda' = \lceil \lambda \frac{(n-1)(n-2)}{(k-1)(k-2)} \rceil + 6(n-1)\lceil \lambda \frac{n-2}{k-2} \rceil + \lambda(\frac{(2n-1)(4n-1)}{3} - 6n + 5)$. *Then* $C_{\lambda'}(4n,4k,3) \leq \frac{(2n-1)(4n-1)}{3} C_\lambda(n,k,3)$ *for every* $n \equiv 1,2$ *(mod* 3*) except possibly for* $n \in P$, *where* P *is defined in Theorem 5.2.*

For the existence of optimal 3-$(n,4,3)$ coverings, the following result can be found, for example, in [10].

Theorem 5.4 *The Schönheim lower bound* $L_\lambda(v,k,t)$ *is defined to be*

$$L_\lambda(v,k,t) = \lceil \frac{v}{k} \lceil \frac{v-1}{k-1} \cdots \lceil \frac{v-t+1}{k-t+1} \lambda \rceil \cdots \rceil \rceil.$$

(1) For any positive integer n *such that* $n \not\equiv 7 \pmod{12}$, $C_1(n,4,3) = L_1(n,4,3)$.

(2) For any $n \equiv 7 \pmod{12}$ *with* $n \geq 52423$, $C_1(n,4,3) = L_1(n,4,3)$; *for* $n = 7$, $C_1(n,4,3) = 12 = L_1(n,4,3) + 1$.

Combining the above Corollary 5.3 and Theorem 5.4, we can have the following consequence.

Corollary 5.5 *Let* $\lambda' = \lceil \frac{(n-1)(n-2)}{6} \rceil + 6(n-1)\lceil \frac{n-2}{2} \rceil + \frac{(2n-1)(4n-1)}{3} - 6n + 5$. *Then* $C_{\lambda'}(4n,16,3) \leq \frac{(2n-1)(4n-1)}{3} L_1(n,4,3)$ *for every* $n \equiv 1,2,4,5,8,10,11 \pmod{12}$ *except possibly for* $n \in P$, *where* P *is defined in Theorem 5.2, and for every* $n \equiv 7 \pmod{12}$ *with* $n \geq 52423$. *For* $n = 7$, $C_{193}(4n,16,3) \leq 1404$.

A degenerate form of Theorem 5.1 is the following Theorem 5.6. For the existence of a resolvable 2-(v,k,λ) design, the reader is referred to [4] for details. It is well known that a resolvable 2-$(2n,2,1)$ design exists for any positive integer n.

Theorem 5.6 *Let* $\lambda'' = 3\lceil \frac{\lambda(n-2)}{k-2} \rceil + (2n-4)\lambda$. *Then* $C_{\lambda''}(2n,2k,3) \leq (2n-1)C_\lambda(n,k,3)$.

Proof Let $(\mathcal{V},\mathcal{B})$ be a resolvable 2-$(2n,2,1)$ design with resolution classes $R_1, R_2, \ldots, R_{2n-1}$, where $R_i = \{B_{i1}, B_{i2}, \ldots, B_{in}\}$ for $i = 1,2,\ldots,2n-1$. We construct an optimal 3-(n,k,λ) covering on $R_i = \{B_{i1}, B_{i2}, \ldots, B_{in}\}$, where each B_{ij} is viewed as a point. Then it can be checked that the resultant set system is a 3-$(2n,2k,\lambda'')$ covering, with a total of $(2n-1)C_\lambda(n,k,3)$ blocks. We are only

to check that every 3-subset of points is contained in at least λ'' blocks. Let $\{x_1, x_2, x_3\}$ be an arbitrary 3-subset of $\mathcal{V}$. If two of the three points form a block in a resolution class of the resolvable 2-$(2n, 2, 1)$ design, then this 3-subset is contained in at least $\lceil \frac{\lambda(n-2)}{k-2} \rceil$ blocks of the optimal 3-(n, k, λ) covering defined on this resolution class. If the three points belong to different blocks in this resolution class, then they are contained in at least λ blocks in the optimal 3-(n, k, λ) covering defined on this resolution class. So totally there are at least λ'' blocks which contain $\{x_1, x_2, x_3\}$. □

Applying Theorem 5.6 with Theorem 5.4, we immediately obtain the following consequence.

Corollary 5.7 *Let* $\lambda'' = 3\lceil \frac{n}{2} \rceil + 2n - 7$. *Then* $C_{\lambda''}(2n, 8, 3) \leq (2n-1)L_1(n, 4, 3)$ *for every* $n \not\equiv 7 \pmod{12}$ *and for every* $n \equiv 7 \pmod{12}$ *with* $n \geq 52423$. *For* $n = 7$, $C_{19}(2n, 8, 3) \leq 156$.

We should note that the resultant 3-(nl, kl, λ') coverings constructed in Theorem 5.1 and 3-$(2n, 2k, \lambda'')$ coverings in Theorem 5.6 are, in general, not optimal. The value $C_\lambda(v, k, t)$ cannot exceed the Schönheim lower bound [14] $L_\lambda(v, k, t)$. It is known (see [16]) that $C_1(6, 4, 3) = L_1(6, 4, 3) = 6$, and $L_{14}(12, 8, 3) = 57$. By Theorem 5.6, we can only know that $C_{14}(12, 8, 3) \leq 66$. But anyhow, we obtained several "good" 3-$(n, n-k, \lambda)$-coverings with reasonably small numbers of blocks, remembering the difficulty of constructing optimal t-$(n, n-k, \lambda)$-coverings with $t \geq 3$. On the other hand, if the ingredient optimal 3-(n, k, λ) covering is indeed a 3-(n, k, λ) design, then the resultant 3-(nl, kl, λ') covering and 3-$(2n, 2k, \lambda'')$ covering are optimal, and in fact are a 3-(nl, kl, λ') design and a 3-$(2n, 2k, \lambda'')$ design, respectively. This assertion was first proved in Ling [8].

6 Conclusion

In this paper, we proposed an algorithm for the problem of determining a consistent set of shares in a threshold scheme with cheaters to reconstruct the secret correctly. This algorithm is a modification of Rees et al. 's [13] two algorithms but can reduce the number of shares to be pooled to $k + t$ in the best case, and less than or equal to $k + 2t$ in other cases, while Rees et al. 's algorithms need all the good shares to be pooled so that the least number of shares is $n - t$ in the best case and n in other cases. The efficiencies of these three algorithms are similar but the shares needed in Rees et al. 's two algorithms are much more than those in ours, especially when n is large. In addition, the number of shares used in the newly proposed algorithm is usually less than $k + 2t$, the number of shares needed in McEliece and Sarwate [11], which happens more often for large n. Therefore, there is a real advantage in using the newly proposed algorithm, especially in the case when gathering all the participants together is impossible, or in a (k, n) threshold scheme with small k and t but a relatively large n. Another advantage is that only one covering, the t-$(k + 2t, 2t, 1)$ covering, is needed in any (k, n_s) multiple threshold scheme of s multiplicity, or in s different (k, n_s)

threshold schemes with the same k and t but different n_s. As a consequence, this modified algorithm is shown to be practical. We also described a construction for optimal or nearly optimal coverings, which can be used in Algorithms 1 and 3.

Acknowledgments This research is supported by Grant-in-Aid for Scientific Research (C) under Contract Number 14540100. An early version of this paper was published in part in ICISC 2003, Lecture Notes in Comput. Sci. **2971**, Springer-Verlag, Berlin, 2004, pp. 377–385. The authors thank Professor Ruizhong Wei of Lakehead University for his helpful comments on an earlier version of this paper.

References

[1] G. R. Blakley, *Safeguarding cryptographic keys*, AFIPS Conf. Proceed. **48** (1979), 313–317.

[2] E. F. Brickell and D. R. Stinson, *The detection of cheaters in threshold schemes*, SIAM J. Discrete Math. **4** (1991), 502–510.

[3] C. J. Colbourn and R. Mathon, *Steiner systems*, in: C. J. Colbourn and J. H. Dinitz, eds., *The CRC Handbook of Combinatorial Designs*, CRC Press, Boca Raton, 1996, 66–75.

[4] S. C. Furino, Y. Miao and J. Yin, *Frames and Resolvable Designs*, CRC Press, Boca Raton, 1996.

[5] L. Harn, *Efficient sharing (broadcasting) of multiple secrets*, IEE Proc. - Comput. Digit. Tech. **142** (1995), 237–240.

[6] S. Kageyama and Y. Miao, *A construction of resolvable designs*, Utilitas Math. **46** (1994), 49–54.

[7] D. L. Kreher, *t-designs, $t \geq 3$*, in: C. J. Colbourn and J. H. Dinitz, eds., *The CRC Handbook of Combinatorial Designs*, CRC Press, Boca Raton, 1996, 47–66.

[8] A. C. H. Ling, *Union constructions for t-designs*, preprint.

[9] W. H. Mills, *Covering designs I: coverings by a small number of subsets*, Ars Combin. **8** (1979), 199–315.

[10] W. H. Mills and R. C. Mullin, *Coverings and packings*, in: J. H. Dinitz and D. R. Stinson, eds., *Contemporary Design Theory*, John Wiley & Sons, New York, 1992, 371–399.

[11] R. J. McEliece and D. V. Sarwate, *On sharing secrets and Reed-Solomon codes*, Comm. ACM **24** (1981), 583–584.

[12] W. Ogata and K. Kurosawa, *Optimum secret sharing scheme secure against cheating*, Lecture Notes in Comput. Sci. **1070** (1996), 200–211.

[13] R. S. Rees, D. R. Stinson, R. Wei and G. H. J. van Rees, *An application of covering designs: determining the maximum consistent set of shares in a threshold scheme*, Ars Combin. **53** (1999), 225–237.

[14] J. Schönheim, *On coverings*, Pacific J. Math. **14** (1964), 1405–1411.

[15] A. Shamir, *How to share a secret*, Comm. ACM **22** (1979), 612–613.

[16] D. R. Stinson, *Coverings*, in: C. J. Colbourn and J. H. Dinitz, eds., *The CRC Handbook of Combinatorial Designs*, CRC Press, Boca Raton, 1996, 260–265.

[17] M. Tompa and H. Woll, *How to share a secret with cheaters*, J. Cryptology **1** (1988), 133–138.

[18] URL: http://www.ccrwest.org/cover.html

Appendix 1

A computer experiment is executed 1,000 times for each instance. The specific set system $\mathcal{T}$ corresponding to an optimal 2-$(n, 5, 1)$ covering is taken from [18], while the set system $\mathcal{T}'$ is defined to be the same as $\mathcal{T}$ except that the first block in $\mathcal{T}$ is repeated 50 times. $\beta_{\mathcal{T}}$ and $\beta_{\mathcal{T}'}$ are the average number of times needed in Algorithm 1 to succeed in reconstructing the secret correctly, and $Var(\beta_{\mathcal{T}})$ and $Var(\beta_{\mathcal{T}'})$ are the variances of $\beta_{\mathcal{T}}$ and $\beta_{\mathcal{T}'}$, respectively.

No.	n	k	t	β_T	$\beta_{T'}$	$Var(\beta_T)$	$Var(\beta_{T'})$
1	6	1	2	1.416	8.450	0.397	175.379
2	7	2	2	1.732	13.668	0.594	261.125
3	8	3	2	2.210	17.706	1.242	301.791
4	9	4	2	2.621	17.493	1.953	288.283
5	10	5	2	3.137	19.790	2.714	291.015
6	11	6	2	3.719	21.576	4.064	305.164
7	12	7	2	4.507	22.372	6.620	315.949
8	13	8	2	5.155	24.562	8.623	340.276
9	14	9	2	6.114	26.505	12.027	342.593
10	15	10	2	6.443	28.877	14.181	367.585
11	16	11	2	7.437	27.996	18.388	371.391
12	17	12	2	8.108	28.524	21.266	388.495
13	18	13	2	8.997	29.529	25.941	399.041
14	19	14	2	9.697	31.258	30.629	418.859
15	20	15	2	10.493	31.380	36.010	427.305
16	21	16	2	10.876	33.242	36.251	440.691
17	22	17	2	13.140	34.895	60.843	516.973
18	23	18	2	13.935	36.540	63.013	542.476
19	24	19	2	15.178	36.907	74.148	514.226
20	25	20	2	15.618	38.864	73.864	554.715
21	26	21	2	18.099	40.317	110.715	651.214
22	27	22	2	19.127	42.359	119.103	688.752
23	28	23	2	19.989	43.604	133.797	686.361
24	29	24	2	21.643	44.842	150.172	705.209
25	30	25	2	23.944	46.633	200.169	845.938
26	31	26	2	25.727	46.432	210.264	842.413
27	32	27	2	25.059	49.894	223.580	905.116

Appendix 2

Common notations

n: the total number of shares.

k: the threshold value of a (k, n) threshold scheme.

t: the number of fake shares in a (k, n) threshold scheme at most.

K: the secret in a (k, n) threshold scheme.

S: the set of all n shares.

S_t: any t-subset of shares in S.

$P_0(x)$: a polynomial of degree at most $k-1$ such that $y_i = P_0(x_i)$ for at least $n-t$ of the n shares.

$P_0(0)$: the secret which can be reconstructed from any k genuine shares.

G: the set of good shares.

B: the set of bad shares.

C_T: the set of shares satisfying the polynomial $P_T(x)$.

NC_T: the set of shares not satisfying the polynomial $P_T(x)$.

Special notations in Algorithm 1

$\mathcal{T}$: a collection of k-subsets of $\{1, 2, \ldots, n\}$ such that its complement forms the collection of blocks of a t-$(n, n-k, 1)$ covering.

T: a random k-subset of $\mathcal{T}$.

Special notations in Algorithm 2

T: a random k-subset of $\{1, 2, \ldots, n\}$.

Special notation in Algorithm 3

$\mathcal{R}$: a random $(k+2t)$-subset of $\{1, 2, \ldots, n\}$.

$S_{\mathcal{R}}$: $\{(x_i, y_i) : i \in \mathcal{R}\} \subseteq S$.

$\mathcal{T}$: a collection of k-subsets of $\mathcal{R}$ such that its complement is the collection of blocks of a t-$(k+2t, 2t, 1)$ covering.

T: a random k-subset of $\mathcal{T}$.

Special notations in Table 1

β_{d_1}: the expected number of iterations of Algorithm 1.

β_r: the expected number of iterations of Algorithm 2.

β_{d_3}: the expected number of iterations of Algorithm 3.

s_{d_1}: the average number of shares needed in Algorithm 1.

s_{d_3}: the average number of shares needed in Algorithm 3.

Special notations in Appendix 1

T': the same as T in Algorithm 1 except that the first block in T is repeated 50 times.

β_T: the average number of times needed in Algorithm 1 with input T to succeed in reconstructing the secret correctly.

$\beta_{T'}$: the average number of times needed in Algorithm 1 with input T' to succeed in reconstructing the secret correctly.

$Var(\beta_T)$: the variances of β_T.

$Var(\beta_{T'})$: the variances of $\beta_{T'}$.