Generalized Secret Sharing and Monotone Functions

Josh Benaloh
University of Toronto

Jerry Leichter
Yale University

Abstract

Secret Sharing from the perspective of *threshold schemes* has been well-studied over the past decade. Threshold schemes, however, can only handle a small fraction of the secret sharing functions which we may wish to form. For example, if it is desirable to divide a secret among four participants A, B, C, and D in such a way that either A together with B can reconstruct the secret or C together with D can reconstruct the secret, then threshold schemes (even with weighting) are provably insufficient.

This paper will present general methods for constructing secret sharing schemes for ***any*** given secret sharing function. There is a natural correspondence between the set of "generalized" secret sharing functions and the set of monotone functions, and tools developed for simplifying the latter set can be applied equally well to the former set.

1 Introduction

The threshold schemes for secret sharing introduced by Blakley ([Blak79]) and Shamir ([Sham79]) have found many applications in recent years. There are, however, many secret sharing applications which do not fit into the model of threshold schemes.

In a recent paper ([ISN87]), Ito, Saito, and Nishizeki describe a general method of secret sharing whereby a secret can be divided among a set P of trustees such that any "qualified subset" of P can reconstruct the secret and such that unqualified subsets cannot. As they point out, it is most sensible to talk only about families of qualified subsets (or *access structures*) $\mathcal{A}$ which satisfy the property

$$A \in \mathcal{A}\ ,\ A \subseteq A' \implies A' \in \mathcal{A}.$$

It is hard to imagine a meaningful method of sharing a secret which does not satisfy this property.

The method of Ito, Saito, and Nishizeki can be roughly described as follows. For each of the (up to order $2^{|P|}$) sets of the access structure $\mathcal{A}$, divide the secret among

each member of the set.[1] Thus, in the worst case, each of the n trustees may have to hold on the order of 2^n shares.

This paper gives a far simpler and more efficient method of developing a secret sharing scheme for any monotone access structure. The idea is to translate the access structure into a monotone formula.

Each variable in the formula is associated with a trustee in P, and the value of the formula is true if and only if the set of variables which are true corresponds to a subset of P which is in the access structure (i.e. the variables which are true correspond to a subset of trustees qualified to reconstruct the secret). This formula is then used as a template to describe how a secret is to be divided into shares.

Since every monotone function can implemented using just AND operators and OR operators, it is sufficient to show how to divide a secret "across" each of these two operators. It will be shown later how these formulae can be made more efficient by using general THRESHOLD operators and appealing to traditional threshold schemes.

Let p_1 and p_2 be trustees in P. To divide a secret s into shares such that p_1 and p_2 can reconstruct s, p_1 can be given a value s_1 and p_2 given a value s_2 such that $s = s_1 + s_2$. If s is selected from the range $0 \leq s < m$, then s_1 and s_2 can be chosen uniformly from this range subject to the constraint that $s = (s_1 + s_2) \bmod m$. In this case it can be shown in a very strong sense that neither p_1 nor p_2 can, without the other, obtain any information whatsoever about s.

To divide a secret s into shares such that p_1 or p_2 can reconstruct s, p_1 and p_2 can simply both be given the value s. With these two building blocks, it is easy to see how to construct a secret sharing scheme for any monotone access structure.

For instance, in the earlier example, a secret sharing scheme is sought for which either A together with B or C together with D can reconstruct the secret value s. The corresponding access structure can be written as $((A \wedge B) \vee (C \wedge D))$. Thus, to share a secret s according to this access structure, the secret is first moved across the OR yielding a situation in which the secret s now must be shared among AB and among CD. The value s is now moved across the two AND operators, yielding shares s_A, s_B, s_C, and s_D belonging respectively to A, B, C, and D such that $s_A + s_B = s$ and $s_C + s_D = s$. If the shares generated when a value is moved across an AND gate are random and independent of other selections, then it is not hard to show in a very strong sense that insufficient subsets of trustees obtain no information whatsoever about the original secret value.

There is, of course, no need to limit these gates to two inputs since both of the above operations generalize directly to gates with arbitrary fan-in. In general, a value can be moved across an arbitrary THRESHOLD operator by appealing to a traditional threshold scheme such as the Shamir scheme ([Sham79]). If some intermediate value s in a formula is to be moved across a threshold operator with n arguments and threshold k, the secret s is divided among the n arguments according to a (k, n)-threshold scheme, and these shares become the intermediate values for the next level of the formula.

[1] There is actually some minimization done as will be described later.

Since AND operators and OR operators are special cases of THRESHOLD operators, it would suffice to apply the Shamir threshold scheme to each operator of the formula. It is, however, often simpler to apply the direct methods above. Although the method of moving a secret across an OR operator described above does correspond exactly to Shamir's method of constructing a $(1, n)$-threshold scheme, the method given of moving a secret across an AND operator is computationally simpler than a Shamir (n, n)-threshold scheme. In addition, the threshold schemes given by Shamir and others have limitations which are not present in the scheme presented here. These limitations will be discussed later.

The method described by Ito, Saito, and Nishizeki in [ISN87] corresponds precisely to the case of minimal CNF-formulae in which conjunctions are formed by use of (n, n)-threshold schemes rather than by simple sums.

It is of course true that every monotone formula can be expressed as a CNF-formula and that there are a great many monotone formulae for which the CNF-formula is the smallest possible representation. However, there are also a great many cases in which the use of general monotone formulae (especially when arbitrary threshold operators are allowed) gives a much smaller formula than the CNF-formula. The number of shares which must be given to each trustee in these schemes as well as the complexity of reconstructing the secret from its shares are directly related to the size of the formula.

2 Preliminaries

To begin with, we must formally define the necessary access structures.

Definition Given a set P, a *monotone access structure* on P is a family of subsets $\mathcal{A} \subseteq 2^P$ such that

$$A \in \mathcal{A},\ A \subseteq A' \subseteq P \quad \Longrightarrow \quad A' \in \mathcal{A}.$$

Definition Let P be a set. The set V of *variables indexed by* P is the set $V = \{v_p : p \in P\}$.

Definition Given a monotone function F on variables indexed by a set P, the *access structure defined by* F is the set of subsets of A of P for which F is true precisely when the variables indexed by A are set to true.

It is clear that for every monotone function F, the access structure defined by F is a monotone access structure.

Definition For a given set P and an monotone access structure $\mathcal{A}$ on P, define $\mathcal{F}(\mathcal{A})$ to be the set of monotone formulae on $|P|$ variables such that for every formula $F \in \mathcal{F}(\mathcal{A})$, the output of F is true if and only if the true variables in F correspond exactly to a set $A \in \mathcal{A}$.

Note that $F, F' \in \mathcal{F}(\mathcal{A})$ implies that F and F' denote the same function. They may, however, represent entirely different formulae to express this function.

3 Generalized Secret Sharing

We can now begin to define secret sharing schemes. We start with a standard definition for threshold schemes.

Definition Given a set S of possible secret values, a (k, n)-threshold scheme on S is a (randomized) method of dividing each $s \in S$ into an array of shares $[s_1, s_2, \ldots, s_n]$ with each $s_i \in S$ such that

1. Given any set of k or more of the s_i, the secret value s is easily reconstructible.
2. Given any set of fewer than k of the s_i, the secret value s is completely undetermined in an information theoretic sense.

Shamir's polynomially based threshold scheme (see [Sham79]) satisfies the above definition whenever $|S|$ is a prime greater than n. It is not hard to remove the restriction that $|S|$ be prime by, for instance, factoring $|S|$ and using Chinese remaindering to encode secrets and shares. This kind of encoding, however, requires that all prime factors of $|S|$ be greater than n.

Other threshold schemes have been suggested by Blakley ([Blak79]), Asmuth and Bloom ([AsBl80]), and Kothari ([Koth84]), for example.

We want to show that no threshold scheme is sufficient to realize secret sharing on general monotone access structures. To do this, we show that there is no threshold scheme (even using weighting or multiple shares) such that the access structure $((A \wedge B) \vee (C \wedge D))$ can be achieved.

Theorem 1 *There exist monotone access structures for which there is no threshold scheme.*

Proof:

Consider the access structure $\mathcal{A}$ defined by the formula

$$((A \wedge B) \vee (C \wedge D)),$$

and assume that a threshold scheme is to be used to divide a secret value s among A, B, C, and D such that only those subsets of $\{A, B, C, D\}$ which are in $\mathcal{A}$ can reconstruct s.

Let a, b, c, and d respectively denote the weight (number of shares) held by each of A, B, C, and D. Since A together with B can compute the secret, it must be the case that $a + b \geq t$ where t is the value of the threshold. Similarly, since C and D can together compute the secret, it is also true that $c + d \geq t$.

Now assume without loss of generality that $a \geq b$ and $c \geq d$. (If this is not the case, the variables can be renamed.) Since $a + b \geq t$ and $a \geq b$, $a + a \geq a + b \geq t$. So $a \geq t/2$. Similarly, $c \geq t/2$. Therefore, $a + c \geq t$.

Thus, A together with C can reconstruct the secret value s. This violates the assumption of the access structure. ∎

Definition For a given threshold scheme, we use $\$_k(s; p_1, p_2, \ldots, p_n)$ to denote the random function which assigns shares $[s_1, s_2, \ldots, s_n]$ of a secret value s to trustees $p_1, p_2, \ldots, p_n$.

For certain access structures, every generalized threshold scheme *must* be able to assign multiple shares to each trustee (see theorem 3). In this case, we use $s_{i,j}$ to denote the j^{th} share given to trustee p_i.

Definition Given a set P and a monotone access structure $\mathcal{A}$ on P, a generalized secret sharing scheme for $\mathcal{A}$ is a method of dividing a secret s into shares $s_{i,j}$ such that

1. When $A \in \mathcal{A}$, the secret s can be reconstructed from the shares $\bigcup_{i \in A} \bigcup_j s_{i,j}$.
2. When $A \notin \mathcal{A}$, the shares $\bigcup_{i \in A} \bigcup_j s_{i,j}$ give (in an information theoretic sense) no information whatsoever about the value of s.

We now define a generalized secret sharing scheme which satisfies the above definition.

Assume that the secret domain S is fixed to be the set $\{0, 1, \ldots, m-1\}$ for some positive integer m. We can now formally define the generalized secret sharing scheme described in section 1.

Let $\$(s, F)$ be the random function for $s \in S$ and a monotone formula F defined as follows.

- $\$(s, v_p)$ assigns the share s to trustee p.
- $\$(s, A \vee B) = \$(s, A) \cup \$(s, B)$.
- $\$(s, A \wedge B) = \$(s_1, A) \cup \$(s_2, B)$, where s_1 and s_2 are uniformly chosen from the secret domain S such that $s = (s_1 + s_2) \bmod m$.

If operators are allowed to have more than two arguments and if THRESHOLD operators are to be used, we add the following.

- $\$(s, \bigvee(F_1, F_2, \ldots, F_n)) = \bigcup_{1 \le i \le n} \$(s, F_i)$
- $\$(s, \bigwedge(F_1, F_2, \ldots, F_n)) = \bigcup_{1 \le i \le n} \$(s_i, F_i)$, where the s_i are chosen uniformly from S such that $s = (\sum_{i=1}^{n} s_i) \bmod m$.
- $\$(s, \text{THRESHOLD}_k(F_1, F_2, \ldots, F_n)) = \bigcup_{1 \le i \le n} \$(s_i, F_i)$, where $\$_k(s; p_1, p_2, \ldots, p_n) = [s_1, s_2, \ldots, s_n]$.

We now show that for every monotone access structure $\mathcal{A}$ and every monotone formula $F \in \mathcal{F}(\mathcal{A})$, the secret sharing scheme defined by $\$(s, F)$ satisfies the definition of a generalized secret sharing scheme.

Theorem 2 *Let P be a set and let $\mathcal{A}$ be a monotone access structure on P. Let F be a member of $\mathcal{F}(\mathcal{A})$, and let s be a secret value in $S = \{0, 1, \ldots, m-1\}$. The secret sharing scheme defined by $\$(s, F)$ is a generalized secret sharing scheme for $\mathcal{A}$.*

Proof:

It is easy to see that for any set $A \in \mathcal{A}$, the shares belonging to the members of A are sufficient to reconstruct the secret value s.

To see that if $A \notin \mathcal{A}$ then the shares belonging to the members of A give no information about the secret value s, we use induction on the number of operators of the formula F.

A formula with no operators consists of a single variable v_p. The access structure defined by v_p is the set of subsets of P which contain the trustee p. Thus, $\$(s, v_p)$ gives the secret value s to p alone and therefore allows only those sets of trustees which include p to determine s.

A monotone formula F with $d > 0$ operators can always be written in the form $\circ(F_1, F_2, \ldots, F_n)$ where $\circ$ is one of $\bigvee$, $\bigwedge$, and THRESHOLD_k, and where each of $F_1, F_2, \ldots, F_n$ is a monotone formula with less than d operators.

If the operator $\circ$ is $\bigvee$, then $\$(s, F_1, F_2, \ldots, F_n)$ is the union over i of $\$(s, F_i)$. By the inductive hypothesis, for each i, the members of a set A of trustees which is not in the access structure $\mathcal{A}$ can obtain no information whatsoever about the value of s from the values of the shares of $\$(s, F_i)$. Since for $i \neq j$, the shares of $\$(s, F_i)$ are chosen completely independently of the shares of $\$(s, F_j)$, no joint information is possible, and therefore, the shares of $\$(s, F)$ held by the members of an A not in $\mathcal{A}$ give no information at all about s.

If the operator $\circ$ is $\bigwedge$, then $\$(s, F_1, F_2, \ldots, F_n)$ is the union over i of $\$(s_i, F_i)$, where the s_i are chosen uniformly according the constraint that $s = (\sum_{i=1}^{n} s_i) \bmod m$. For each set A of trustees not in $\mathcal{A}$, there must be some i such that the shares of $\$(s_i, F_i)$ held my members of A give no information about s_i. (If this were not the case, then A would be in the access structure $\mathcal{A}$.) Since the shares given in each sub-formula are independent, this implies that the sum $s = (\sum_{i=1}^{n} s_i) \bmod m$ is completely undetermined by the shares held by the members of A.

Finally, if the operator $\circ$ is THRESHOLD_k, then $\$(s, F_1, F_2, \ldots, F_n)$ is the union over i of $\$(s_i, F_i)$, where the s_i are assigned according to the threshold scheme $\$_k$ by $\$_k(s; F_1, F_2, \ldots, F_n) = [s_1, s_2, \ldots, s_n]$. By assumption, a threshold scheme $\$_k$ allows sets of fewer than k shareholders to obtain no information at all about the value of s. If A is a set of trustees not in $\mathcal{A}$, then the members of A can obtain direct information about fewer than k of the s_i. Again by independence, the shares held by the members of A provide no information whatsoever about the value of s. ∎

Finally, we show that there are access structures which cannot be realized without giving multiple (or extra large) shares to some trustee.

Theorem 3 *There exists access structures for which any generalized secret sharing scheme must give some trustee shares which are from a domain larger than that of the secret.*

Proof:

Consider the access structure $\mathcal{A}$ defined by the formula

$$((A \wedge B) \vee (B \wedge C) \vee (C \wedge D)),$$

and fix a value a to be the share held by A.

Let $b_1, \ldots, b_m$ represent the set of possible shares available to B. Since A and B are together sufficient to compute the secret value s, each share b_i determines exactly one possible value s_i of the secret s. Also, since the share a alone is insufficient to give any information about the secret value s and since the number of possible values of s is equal to m (the number of possible values of the share held by B), every possible secret value s_i is determined by a together with exactly one value b_i.

Thus, for each a, we can construct a set of m pairs (s_i, b_i) which are consistent with a and such that each possible value of the secret and each possible value of B's share appear in *exactly* one such pair.

Now consider the possible value of the share held by C. Since B and C are together sufficient to compute the secret value s, and since each b_i can be matched with exactly one value c_i to form the secret s_i, there is exactly one value c_i consistent with each pair (s_i, b_i) in the set. (Note that the c_i are *not* necessarily distinct.)

If any two of these c_i are distinct, then considering the value held by A together with the value held by C would eliminate at least one of the possible consistent pairs and thereby eliminate at least one of the possible values of the secret s. But A and C are together *not* sufficient to determine *any* information about the value of the secret s. Thus, the value held by C must be completely determined by the value held by A.

Now since C and D are together sufficient to compute the secret value s, the value held by C together with the value held by D is sufficient to compute the secret value s. However, the value held by A completely determines the value held by C. Thus, the value held by A together with the value held by D is sufficient to compute s. This violates the premise that A and D are insufficient. ∎

4 Generalized Secret Sharing Homomorphisms

In [Bena86] and [Bena87], Benaloh describes a homomorphism property that is present in many threshold schemes which allows shares of multiple secrets to be combined to form "composite shares" which are shares of a composition of the secrets. Such secret sharing homomorphisms also apply to the generalized secret sharing scheme presented here.

For instance, if the shares of a secret value x (drawn from a fixed secret domain $S = \{0, 1, \ldots, m-1\}$) are added to the corresponding shares of a similarly chosen secret value y, then the sums represent shares of the value $(x + y) \bmod m$.

The applications of secret sharing homomorphisms includes fault-tolerant verifiable secret-ballot elections as well as verifiable secret sharing. The methods of verifiable secret sharing developed for threshold schemes in [Bena86] and [Bena87] and also by Feldman in [Feld87] can be used for generalized secret sharing too. The approach used by Feldman is actually somewhat better suited to these purposes. Here, the secret is distributed in such a way as to enable each trustee to, without further interaction, verify that its share is a well-formed and valid share of the secret.

The main requirement of these schemes is the presence of an appropriate homomorphism property, and the homomorphism property described above turns out to be sufficient.

5 Conclusions

This paper has shown how generalized secret sharing can be achieved in a method which is simpler and more efficient than in any previous scheme. There are, however, many cases in which this method is still unable to be applied efficiently.

For any given polynomial P, the number of n-variable monotone formulae of size no more than $P(n)$ is exponential in $P(n)$. However, the total number of monotone functions on n variables is doubly exponential in n. Therefore, most monotone access structures cannot be realized with a polynomially large number of polynomially sized shares.

Further methods of secret sharing which can efficiently realize additional access structures and an analysis of precisely what access structures can be efficiently realized are interesting areas for future research.

Acknowledgements

The authors would like to express their thanks to Ernie Brickell and to Steven Rudich for helpful discussions regarding this work.

References

[AsBl80] **Asmuth, C.** and **Bloom, J.** "A Modular Approach to Key Safeguarding." *Texas A&M University, Department of Mathematics,* College Station, TX (1980).

[Bena86] **Benaloh, J.** "Secret Sharing Homomorphism: Keeping Shares of a Secret Secret" *Proc. Crypto '86,* Santa Barbara, CA (Aug. 1986), 251–260. Published as *Advances in Cryptology,* ed. by A. Odlyzko in *Lecture Notes in Computer Science,* vol. 263, ed. by G. Goos and J. Hartmanis. Springer-Verlag, New York (1987).

[Bena87] **Benaloh, J.** "Verifiable Secret-Ballot Elections." Ph.D. Thesis presented at Yale University (Sep. 1987).

[Blak79] **Blakley, G.** "Safeguarding Cryptographic Keys." *Proc. AFIPS 1979 National Computer Conference,* New York, NY (June 1979), 313–317.

[Feld87] **Feldman, P.** "A Practical Scheme for Non-interactive Verifiable Secret Sharing." *Proc.* 28th *IEEE Symp. on Foundations of Computer Science,* Los Angeles, CA (Oct. 1987), 427–437.

[ISN87] **Ito, M., Saito, A.,** and **Nishizeki, T.** "Secret Sharing Scheme Realizing General Access Structure." *Proc. Glob. Com,* (1987).

[Koth84] **Kothari, S.** "Generalized Linear Threshold Scheme." *Proc. Crypto '84,* Santa Barbara, CA (Aug. 1984), 231–241. Published as *Advances in Cryptology,* ed. by G. Blakley and D. Chaum in *Lecture Notes in Computer Science,* vol. 196, ed. by G. Goos and J. Hartmanis. Springer-Verlag, New York (1985).

[Sham79] **Shamir, A.** "How to Share a Secret." *Comm. ACM 22,* 11 (Nov. 1979), 612–613.