Secret Sharing Limitations Over Boolean Circuits

Iulian Oleniuc, Alexandru Ioniță

Abstract

This paper offers a gentle introduction into the realm of monotone span programs and their connection with linear secret sharing schemes and attribute-based encryption while emphasizing the cryptographic importance of finding efficient MSPs for representing complex access structures. We provide a proof that there is no ideal LSSS for Boolean circuits, thus tackling the open problem of finding LSSSes of non-exponential size for Boolean circuits. Moreover, we present an application of our proof to graph access structures and a backtracking approach to finding efficient MSPs for given access structures.

Keywords: Cryptography, Attribute-Based Encryption, Linear Secret Sharing, Monotone Span Programs, Boolean Circuits.

MSC 2020: 94A60, 68P25, 94A62.

1 Introduction

Modern enterprise software relies more and more on cloud services for in-common file storage and collaborative access to data. These systems bring up a crucial privacy problem. Usually, the cloud service provider has access to all the sensitive data.

The paradigm of attribute-based encryption (ABE) comes as a solution since it assigns attributes to both users and files. The authorization of a user to perform some action on a specific file is made by evaluating the attributes involved. Thus, this type of encryption lets us define complex fine-grained access policies based on the relations between the values of different attributes (of either the user or the file).

©2025 by Computer Science Journal of Moldova
doi:10.56415/csjm.v33.06

Goyal et al. [1] proposed the first ABE construction, relying on bilinear maps and threshold trees as access structures. Many other schemes have been proposed, with various flavors, such as *access revocation* [2], [3], *decentralized settings* [4], and *outsourced decryption* [5].

A frequently studied problem in ABE is the construction of efficient systems with expressive access structures. While the first ABE system [1] has a certain degree of expressivity, using more advanced access structures, such as Boolean circuits, is problematic in the bilinear map setting. All state-of-the-art schemes for such access structures yield exponentially large keys in order to provide secure constructions [6], [7].

Garg et al. [8] constructed the first ABE system for Boolean circuits relying on multilinear maps. However, for these cryptographic primitives, there are no known secure cryptographic constructions [9], [10]. Garg also conjectured that ABE for Boolean circuits cannot be obtained from bilinear maps (in an efficient manner).

1.1 Our Contribution

We have addressed the open problem of constructing secure and efficient ABE schemes for Boolean circuits. Since the key step in this issue is the secret sharing scheme used in the key generation procedure in ABE, we have focused on *linear secret sharing schemes* (LSSSes) over Boolean circuits.

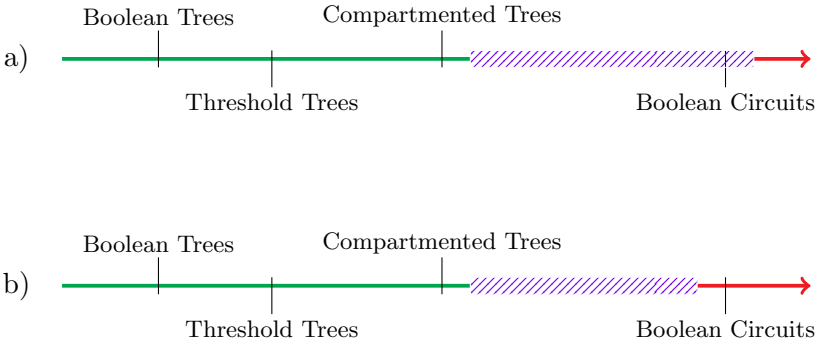

Figure 1. Expressiveness scale for access structure classes

Figure 1 (a) puts the four access structure classes defined in Section 3.1, and whose connection with MSPs is described in Section 2.2, on an expressiveness scale. The green segment represents classes for which there are known ideal LSSSes, whereas the red segment represents classes that are only known to admit non-ideal LSSSes. The purple section is the gap to be tightened. That is, we want to find new access structure classes that either are more expressive than compartmented trees and admit ideal LSSSes or that do not admit ideal LSSSes at all.

We provide the first proof that *ideal* LSSSes cannot be constructed to represent *monotone* Boolean circuits in a general manner. We do this by using the equivalence between LSSSes and monotone span programs (MSPs) and proving that no MSP exists for a particular circuit. Thus, the scale of expressiveness slightly changes to Figure 1 (b).

Moreover, we provide a framework [11] for brute-force searching MSPs for given access structures.

2 Related Work

2.1 Linear Secret Sharing Schemes

Secret sharing is a field that received a tremendous amount of attention in the last half a century. Threshold secret sharing is probably the most studied variation, for which many ideal schemes have already been achieved. They are based on Lagrange interpolation [12], linear algebra [13], and the Chinese remainder theorem [14], [15], among others. From our knowledge, there are no secret sharing schemes for Boolean circuits other than the ones that can be extracted from the schemes proposed for ABE constructions [6], [7]. However, these schemes can lead to exponential key size in the number of parties. That is, they are not ideal.

2.2 Monotone Span Programs

Linear secret sharing schemes (LSSSes) and monotone span programs (MSPs) – a model of computation based on linear algebra – were well-studied by Beimel, especially throughout his PhD thesis [16].

The correlation between MSPs, LSSSes, and Boolean circuits has been studied in various combinations and for various access structures. However, the existing literature on these problems is unorganized and studies these concepts in isolation from one another. Also, there is a lack of correlation between circuit theory and MSPs.

For example, take the construction of [17] for NonBipartite_m access structure, which only needs m rows when working in GF(2). The input of this function is a binary array of length $m = \binom{n}{2}$, encoding the edges of a graph G with n vertices, and its output is 1 if and only if G is not bipartite. The trivial construction of a Boolean circuit representing NonBipartite_m access structures will result in a circuit of exponential size in the number of parties, and therefore, not of use in our problem.

Another such function [18] is PerfectMatching_n, which tests whether a graph contains a perfect matching. An MSP can compute this function using a number of rows that is polynomial in n. These lower bounds are not relevant in our context, since, again, transposing a graph access structure into a Boolean circuit is not trivial.

Constructions Various operations over MSPs were developed [19] to help in designing conversions from particular access structure classes (e.g., Boolean trees) into MSPs. There are known MSP constructions starting from Boolean trees of in-degree 2 [20, Appendix G] and from threshold trees [21].

Lower Bounds Wegener authored a comprehensive study on the complexity of Boolean functions [22], which comes in handy when it is to decide what specific access structure class should be used to represent a particular Boolean function. Moreover, some lower bounds for the minimum size of MSPs computing peculiar Boolean functions were proven in the literature [17], [18].

2.3 Attribute-Based Encryption

Attribute-based encryption (ABE) is a public-key encryption paradigm, which comes in two flavors, namely ciphertext-policy (CP-ABE) and

key-policy (KP-ABE). The latter was implemented using many different types of cryptographic primitives, such as lattices [23] and quadratic residues [24], the one of interest for us being *bilinear pairings*.

The cornerstone paper of pairing-based KP-ABE [1] presents an ideal scheme for threshold trees, as well as an efficient and more general one, which works for any LSSS-realizable access structure, following the same main idea. An ideal scheme for compartmented trees [25] exists too.

One open problem is to construct an efficient ABE system for Boolean circuits using bilinear maps. The existing schemes on this topic have exponential expansion in the key size [6], [7]. However, recent works try to improve these results, either by directly optimizing the circuit structure using heuristics [26], or by constructing more efficient secret sharing schemes for some particular Boolean circuits, such as compartmented groups [25].

3 Preliminaries

3.1 Access Structures

Secret sharing is a cryptographic technique used to distribute a secret among a group of parties in such a way that only specific subsets of parties, called *authorized* subsets, can reconstruct the secret.

In the context of secret sharing, the set of authorized subsets forms what is called an *access structure*.

Definition 1 (Monotone Access Structure)**.** *Let $\mathcal{P}$ be the set of parties. A collection $\mathbb{A} \subseteq 2^{\mathcal{P}}$ is called a monotone access structure if $\mathcal{X} \in \mathbb{A} \wedge \mathcal{X} \subseteq \mathcal{Y} \rightarrow \mathcal{Y} \in \mathbb{A}$.*

Hereinafter, we shall refer to monotone access structures simply as "access structures."

Definition 2 (Boolean Circuit)**.** *A Boolean circuit is a directed acyclic graph (DAG) such that, for each node v in the graph, either v has in-degree 0 and is labeled with a party $X \in \mathcal{P}$ that no other such node is labeled with, or v has in-degree $n > 1$ and is labeled with one of the $\wedge$ (AND) and $\vee$ (OR) Boolean operators.*

Definition 3 (Boolean Tree)**.** *A Boolean circuit where each gate has an out-degree less than or equal to* 1 *is called a Boolean tree. It can also be viewed as the abstract syntax tree (AST) of a logical formula containing only the* $\wedge$ *and* $\vee$ *operators (and no negations).*

Definition 4 (Threshold Tree)**.** *A threshold tree (referred to as "access tree" in [1]) only uses threshold gates as operators. A threshold gate is characterized by the threshold* t *and is satisfied only when at least* t *of its input nodes are also satisfied.*

For example, $(3/ABCDE)$ is true only when at least three literals from $\{A, B, C, D, E\}$ are true.

Any Boolean tree can be simulated by a threshold tree, since an $\wedge$ gate of in-degree n can be replaced by an n-threshold gate and an $\vee$ gate of in-degree n can be replaced by a 1-threshold gate.

Definition 5 (Compartmented Tree)**.** *A compartmented tree (referred to as "CAS-tree" in [25]) only uses compartmented gates as operators. A compartmented gate divides its* n *input wires into* k *disjoint compartments, such that* $n = n_1 + \cdots + n_k$*, where* n_i *is the number of input wires of compartment* i*. Each compartment* i *has a threshold* $t_i \leq n_i$*, while the entire gate also has its own threshold* t*, such that* $t_1 + \cdots + t_k \leq t \leq n$*. Indeed, a compartmented gate is satisfied only when all its* $k + 1$ *thresholds are satisfied.*

For example, $(3; 1, 1/AB, CDE)$ is true only when at least one literal from $\{A, B\}$ is true, at least one literal from $\{C, D, E\}$ is true, and at least three literals from $\{A, B, C, D, E\}$ are true.

Obviously, any threshold tree can be simulated by a compartmented tree, since a threshold gate can be replaced by a similar compartmented gate of only one compartment.

3.2 Linear Secret Sharing Schemes

Definition 6 (Linear Secret Sharing Scheme [16])**.** *Let* K *be a finite field and* Π *be a secret sharing scheme with domain of secrets* $S \subseteq K$ *realizing an access structure* $\mathbb{A}$*. We say that* Π *is a linear secret sharing scheme over* K *if:*

1. *The piece of each party is a vector over K. That is, for every i, there exists a constant d_i such that the piece of P_i is taken from K^{d_i}. We denote by $\Pi_{i,j}(s,r)$ the jth coordinate in the piece of P_i, where $s \in S$ is a secret and $r \in R$ is the random input given by the dealer.*

2. *For every authorized set, the reconstruction function of the secret from the pieces is linear. That is, for every $G \in \mathbb{A}$, there exist constants $\{\alpha_{i,j} \mid P_i \in G, 1 \leq j \leq d_i\}$ such that, for every secret $s \in S$ and every choice of random input $r \in R$,*

$$s = \sum_{P_i \in G} \sum_{1 \leq j \leq d_i} \alpha_{i,j} \cdot \Pi_{i,j}(s,r),$$

where the constants and the arithmetic are over K.

The total size of the pieces in the scheme is defined as $d = \sum_{i=1}^{n} d_i$.

Intuitively, in an LSSS, each share is given in the form of one or multiple vectors and the reconstruction of the secret involves only linear operations over these vectors. The size of each individual vector equals the size of the secret.

Definition 7 (Ideal LSSS)**.** *An LSSS is said to be ideal if each share consists of exactly one vector.*

3.3 Monotone Span Programs

Definition 8 (Monotone Span Program [16])**.** *Let K be a finite field and $\mathcal{P}$ be the set of parties. A monotone span program over K is a labeled matrix $\hat{M}(M,\rho)$, where M is an $m \times d$ matrix over K and ρ is a labeling of the rows of M by literals from $\mathcal{P}$.*

For every input $\delta \in \{0,1\}^n$, let M_δ be the submatrix of M consisting of the rows whose labels are set to 1 by δ. The monotone span program $\hat{M}$ accepts δ if $\vec{1} \in \text{span}(M_\delta)$. A monotone span program computes an access structure $\mathbb{A}$ if it only accepts the inputs δ that encode authorized sets from $\mathbb{A}$.

The vector $\vec{1} = (1, \ldots, 1)$ is called the *objective* vector. In the above definition, $\vec{1}$ can be replaced by any other non-zero vector via a change of basis. Usually, $(1, 0, \ldots, 0)$ is used instead.

Definition 9 (MSP Size)**.** *The size of an $m \times d$ MSP is defined simply as its number of rows m.*

Theorem 1 ([27])**.** *It is always possible to restrict the matrix of an MSP to a set of linearly independent columns without changing the function computed by the program. Therefore, it is not necessary to use more columns than rows and hence the above definition of size.*

Theorem 2 ([16])**.** *Let $\mathbb{A}$ be an access structure and K be a finite field. There exists an MSP of size d over K computing $\mathbb{A}$ if and only if there exists an LSSS over K realizing $\mathbb{A}$ in which the total size of the pieces is d.*

4 Linear Secret Sharing in Boolean Circuits

In the literature, there exists an efficient MSP construction for Boolean trees of in-degree 2 [20], but it has not been correlated yet with *general* access structures. Therefore, this section presents the entire flow of converting access structures to MSPs.

The most straightforward approach to constructing an MSP from a given access structure begins by viewing $\mathbb{A}$ as a Boolean formula in disjunctive normal form (DNF). In fact, this is the most natural way of describing $\mathbb{A}$ – an authorized subset $\mathcal{X} = \{X_1, \ldots, X_n\}$ becomes a minterm $X_1 \wedge \cdots \wedge X_n$. Then, this Boolean formula is rewritten as a tree of in-degree 2. Finally, this tree is given as input to the standard MSP conversion procedure [20, Appendix G].

The downside of this approach is that the size of the MSP is equal to the number of literals in the DNF formula, which sometimes can be exponential in the number of parties. For instance, it is a well-known fact that the conjunctive normal form (CNF) formula $(X_1 \vee Y_1) \wedge \cdots \wedge (X_n \vee Y_n)$ requires an exponential number of literals in order to be rewritten in DNF. Therefore, the access structure induced by it will generate a very inefficient MSP.

As we have mentioned before, Garg et al. [8] conjectured that it is impossible to construct ABE schemes from bilinear maps supporting monotone Boolean circuit access structures. We stress that their claim must be interpreted as "it is impossible to construct *efficient* ABE schemes from bilinear maps supporting monotone Boolean circuits." We prove part of this claim, more exactly, we prove that there is no LSSS for Boolean circuits. We stress that in order to build an ABE system for Boolean circuits, there must exist a secret sharing mechanism through the access structure, which is used in the key generation procedure.

4.1 Boolean Circuits Do Not Admit Ideal LSSSes

Theorem 3. *There is no ideal linear secret sharing scheme for the class of access structures represented by monotone Boolean circuits.*

Proof. In order to prove that Boolean circuits do not always support ideal LSSSes, and since, according to Theorem 2, MSPs and LSSSes are equivalent, we prove that MSPs of size $|\mathcal{P}|$ cannot be constructed for such access structures.

Our proof is based on a counterexample. We take the specific access structure $\mathbb{A} = \{\{A, B\}, \{B, C\}, \{C, D\}\}$ and prove that it does not admit any MSP of size $|\mathcal{P}|$. We mention that, obviously, $\mathbb{A}$ cannot be viewed as a Boolean tree, but it *can* be viewed as a Boolean circuit, like in Figure 2.

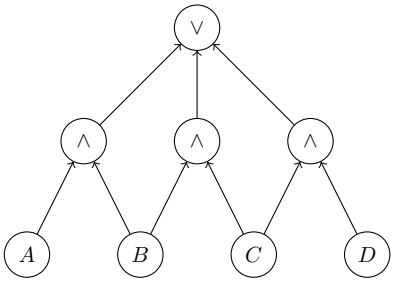

Figure 2. Access structure $\{\{A, B\}, \{B, C\}, \{C, D\}\}$ represented as a Boolean circuit

Let us assume $\mathbb{A}$ admits an MSP with

$$\rho = \{(1, A), (2, B), (3, C), (4, D)\}$$

and let its rows be $\vec{A}$, $\vec{B}$, $\vec{C}$, and $\vec{D}$, respectively. Thus, there exist $a, b, c, d, e, f \in \mathbb{Z}_p$ such that

$$\begin{aligned} a\vec{A} + b\vec{B} &= \vec{1}, \\ c\vec{B} + d\vec{C} &= \vec{1}, \\ e\vec{C} + f\vec{D} &= \vec{1}. \end{aligned}$$

First, notice that no coefficient can be zero. Take, for instance, $a = 0$. Then, from the first equation it would result that $b\vec{B} = \vec{1}$. This implies that $\{B\} \in \mathbb{A}$, which violates the hypothesis.

Multiplying the first equation by c, the second one by b, and then subtracting them leads to

$$ac\vec{A} - bd\vec{C} = (c - b)\vec{1}.$$

As explained above, it is safe to divide this by bd. After some algebraic manipulation, we get

$$ace\vec{A} + bdf\vec{D} = (bd + ce - be)\vec{1}.$$

First Case If $bd + ce - be \neq 0$, then

$$\frac{ace}{bd + ce - be}\vec{A} + \frac{bdf}{bd + ce - be}\vec{D} = \vec{1},$$

and therefore $\{A, D\} \in \mathbb{A}$, which cannot be true.

Second Case If $bd + ce - be = 0$, then

$$\vec{A} = -\frac{bdf}{ace}\vec{D}.$$

Now, in any equation, $\vec{A}$ can be replaced by $(-bdf/ace)\vec{D}$ and $\vec{D}$ can be replaced by $(-ace/bdf)\vec{A}$. Therefore, the first equation (i.e., $\{A, B\} \in \mathbb{A}$) implies that $\{D, B\} \in \mathbb{A}$ and the third one (i.e., $\{C, D\} \in \mathbb{A}$) implies that $\{C, A\} \in \mathbb{A}$. Both findings violate the given access structure. □

4.2 MSPs and Graph Access Structures

As an application of the ideas presented in the previous proof, we look into graph access structures and we provide a necessary condition for them to admit ideal LSSSes. This condition was already proved to also be *sufficient* [28].

Definition 10 (Graph Access Structure)**.** *Let $\mathcal{P}$ be the set of parties and let $\mathcal{A}$ be an access structure over $\mathcal{P}$. If $\mathbb{A} \subseteq \{\{U, V\} \mid U, V \in \mathcal{P}, U \neq V\}$, then $\mathcal{A}$ is a graph access structure.*

Lemma 1. *Let $A, B, C \in \mathcal{P}$. If $\{A, B\} \in \mathbb{A}$, $\{B, C\} \in \mathbb{A}$, $\{A, C\} \notin \mathbb{A}$, and $\{X \mid \{A, X\} \in \mathbb{A}\} \neq \{X \mid \{C, X\} \in \mathbb{A}\}$ (i.e., the adjacency lists of A and C are different), then $\mathbb{A}$ does not admit an ideal LSSS.*

Proof. Some details were already touched in the previous proof, and thus are ommited. Since sets $\{A, B\}$ and $\{B, C\}$ are authorized, then there exist $a, b, c, d \in \mathbb{Z}_p^*$ such that

$$a\vec{A} + b\vec{B} = \vec{1},$$
$$c\vec{B} + d\vec{C} = \vec{1}.$$

Through some algebraic manipulation we get

$$ac\vec{A} - bd\vec{C} = (c - b)\vec{1}.$$

Case $b \neq c$ would lead to $\{A, C\} \in \mathbb{A}$, which is false. Thus, $b = c$ and, consequently, $\vec{A} = (bd/ac)\vec{C}$. Therefore, any $\{A, X\} \in \mathbb{A}$ also implies that $\{C, X\} \in \mathbb{A}$, and any $\{C, X\} \in \mathbb{A}$ also implies that $\{A, X\} \in \mathbb{A}$. □

Lemma 1 shows that, if a graph $\mathbb{A}$ contains any subgraph $\{A, B, C\}$ with said property, then $\mathbb{A}$ does not admit an ideal LSSS. The full implication of this result [28] is that a graph admits an ideal LSSS if and only if it is multipartite. That being said, graph access structures are obviously less expressive than compartmented trees, and so this result may not be that relevant for our goals.

4.3 Backtracking MSP Construction

In our search for an LSSS for Boolean circuits, we created a tool for generating MSPs of specified size for certain access structures. We made this tool publicly available [11]. We continue with a brief description of our tool.

In order to obtain better MSPs, of arbitrarily small size, we tried a backtracking approach that generates candidates for matrix M. The inputs are:

- $\mathbb{A}$ – the given access structure;
- $m \times d$ – the dimensions of the matrix;
- ρ – the labeling of the rows of the matrix;
- p – the matrix is defined over $\mathbb{Z}_p$;
- k – the elements of the matrix take values from $\{0, 1, \ldots, k-1\}$.

The matrix is generated row by row and element by element. When a row i is finished, every submatrix M_I induced by a subset of rows $I \subseteq \{1, 2, \ldots, i\}$, with $i \in I$, is analyzed in order to abort this branch of execution if possible. Thus, the following Boolean values are computed:

- `full` – there is no row $j \notin I$ with $\rho(j) \in \{\rho(i) \mid i \in I\}$;
- `auth` – $\{\rho(i) \mid i \in I\}$ is an authorized subset;
- `span` – $\vec{1} \in \text{span}(M_I)$.

Therefore, we can backtrack when `auth` $\wedge$ `full` $\wedge \neg$`span` or $\neg$`auth` $\wedge$ `span`.

We mention that testing if $\vec{1} \in \text{span}(M_I)$ is done using the Rouché-Capelli theorem. That is, M_I spans $\vec{1} = (1, \ldots, 1)$ if and only if

$$\text{rank}(M_I) = \text{rank}\left(\begin{bmatrix} M_I \\ \vec{1} \end{bmatrix}\right).$$

Moreover, if the set is authorized but it is not a minterm (i.e., it is a superset of an authorized set), then we may skip the entire step.

According to our tests, the backtracking algorithm can easily prove that a given access structure, with a small number of literals (i.e., up to 5), does not admit ideal LSSSes with values less than a very small k (i.e., up to 5).

5 Conclusions and Future Work

We have addressed in this paper an open problem regarding secret sharing over Boolean circuits. We have proved that it is impossible to construct ideal LSSSes for monotone Boolean circuits in a general manner. Our proof does not exclude the possibility that polynomial LSSSes may exist for this class of access structures. Thus, there remains an open question of whether such construction exists. However, we think this construction does not exist, and we provide the intuition behind our reasoning.

We will introduce a special type of gate, which we will further refer to as `ABBCCD`. This gate models the access structure $\mathbb{A} = \{\{A, B\}, \{B, C\}, \{C, D\}\}$, meaning that it has four input wires, labeled A, B, C, and D, and one output wire. This gate returns 1 if and only if $(A \wedge B) \vee (B \wedge C) \vee (C \wedge D)$ is true. Now consider the class of access structures represented by a tree consisting of `ABBCCD`-gates as internal nodes. Our intuition is that this tree requires an MSP of exponential size in the total number of inputs. However, we do not have a formal proof for this claim.

Therefore, whether it is possible to achieve MSPs – and, consequently, LSSSes – of non-exponential size for monotone Boolean circuits remains an open problem.

References

[1] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM conference on Computer and communications security*, 2006, pp. 89–98.

[2] A. Boldyreva, V. Goyal, and V. Kumar, "Identity-based encryption with efficient revocation," in *Proceedings of the 15th ACM conference on Computer and communications security*, 2008, pp. 417–426.

[3] X. Liang, R. Lu, X. Lin, and X. S. Shen, "Ciphertext policy attribute based encryption with efficient revocation," *TechnicalReport, University of Waterloo*, vol. 2, p. 8, 2010.

[4] M. Chase, "Multi-authority attribute based encryption," in *Theory of Cryptography Conference*. Springer, 2007, pp. 515–534.

[5] M. Green, S. Hohenberger, B. Waters *et al.*, "Outsourcing the decryption of abe ciphertexts." in *USENIX security symposium*, vol. 2011, no. 3, 2011.

[6] P. Hu and H. Gao, "A key-policy attribute-based encryption scheme for general circuit from bilinear maps." *Int. J. Netw. Secur.*, vol. 19, no. 5, pp. 704–710, 2017.

[7] F. L. Țiplea and C. C. Drăgan, "Key-policy attribute-based encryption for boolean circuits from bilinear maps," in *Cryptography and Information Security in the Balkans: First International Conference, BalkanCryptSec 2014, Istanbul, Turkey, October 16-17, 2014, Revised Selected Papers 1*. Springer, 2015, pp. 175–193.

[8] S. Garg, C. Gentry, S. Halevi, A. Sahai, and B. Waters, "Attribute-based encryption for circuits from multilinear maps," in *Annual Cryptology Conference*. Springer, 2013, pp. 479–499.

[9] M. Albrecht and A. Davidson, "Are graded encoding scheme broken yet," 2017.

[10] F. L. Țiplea, "Multi-linear maps in cryptography," in *Conference on Mathematical Foundations of Informatics*, 2018, pp. 241–258.

[11] I. Oleniuc, "Backtracking algorithm for finding msps for access structures," `https://github.com/gareth618/mspbkt`, 2024.

[12] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[13] G. R. Blakley, "Safeguarding cryptographic keys," in *Managing requirements knowledge, international workshop on*. IEEE Computer Society, 1979, pp. 313–313.

[14] M. Mignotte, "How to share a secret," in *Cryptography: Proceedings of the Workshop on Cryptography Burg Feuerstein, Germany, March 29–April 2, 1982 1.* Springer, 1983, pp. 371–375.

[15] C. Asmuth and J. Bloom, "A modular approach to key safeguarding," *IEEE transactions on information theory*, vol. 29, no. 2, pp. 208–210, 1983.

[16] A. Beimel *et al.*, "Secure schemes for secret sharing and key distribution," 1996.

[17] A. Beimel, A. Gál, and M. Paterson, "Lower bounds for monotone span programs," *Computational Complexity*, vol. 6, pp. 29–45, 1996.

[18] L. Babai, A. Gál, and A. Wigderson, "Superpolynomial lower bounds for monotone span programs," *Combinatorica*, vol. 19, no. 3, pp. 301–319, 1999.

[19] V. Nikov and S. Nikova, "New monotone span programs from old," *Cryptology ePrint Archive*, 2004.

[20] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," in *Annual international conference on the theory and applications of cryptographic techniques.* Springer, 2011, pp. 568–588.

[21] Z. Liu, Z. Cao, and D. S. Wong, "Efficient generation of linear secret sharing scheme matrices from threshold access trees," *Cryptology ePrint Archive*, 2010.

[22] I. Wegener, *The complexity of Boolean functions.* John Wiley & Sons, Inc., 1987.

[23] W. Dai, Y. Doröz, Y. Polyakov, K. Rohloff, H. Sajjadpour, E. Savaş, and B. Sunar, "Implementation and evaluation of a lattice-based key-policy abe scheme," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 5, pp. 1169–1184, 2017.

[24] B. Chandrasekaran and R. Balakrishnan, "Attribute based encryption using quadratic residue for the big data in cloud environment," in *Proceedings of the International Conference on Informatics and Analytics*, 2016, pp. 1–4.

[25] A. Ioniță, "Optimizing attribute-based encryption for circuits using compartmented access structures," *Cryptology ePrint Archive*, 2023.

[26] A. Ioniță, D.-A. Banu, and I. Oleniuc, "Heuristic optimizations of boolean circuits with application in attribute-based encryption," *Procedia Computer Science*, vol. 225, pp. 3173–3182, 2023.

[27] A. Gál, "A characterization of span program size and improved lower bounds for monotone span programs," in *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, 1998, pp. 429–437.

[28] E. F. Brickell and D. M. Davenport, "On the classification of ideal secret sharing schemes," *Journal of Cryptology*, vol. 4, no. 2, pp. 123–134, 1991.

Iulian Oleniuc, Alexandru Ioniță

Received July 24, 2024
Accepted August 23, 2024

Iulian Oleniuc
Alexandru Ioan Cuza University of Iași
11 Carol I Boulevard, Iași 700506, Romania
E–mail: iulian.oleniuc@gmail.com

Alexandru Ioniță
ORCID: https://orcid.org/0000-0002-9876-6121
Alexandru Ioan Cuza University of Iași;
Simion Stoilow Institute of Mathematics of the Romanian Academy
11 Carol I Boulevard, Iași 700506, Romania
E–mail: alexandru.p.ionita@gmail.com