IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 8, 2019, accepted March 25, 2019, date of publication April 1, 2019, date of current version April 16, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2908785

A Quantum Algorithm for the Commutativity of Finite Dimensional Algebras

ELÍAS F. COMBARRO, JOSÉ RANILLA, AND IGNACIO F. RÚA
Departamento de Informática, University of Oviedo, Oviedo, Spain

Corresponding author: José Ranilla (ranilla@uniovi.es)

This work was supported in part by the MINECO under Grant MTM-2017-83506-C2-2-P and Grant MINECO-16-TEC2015-67387-C4-3-R, and in part by the MICINN under Grant RTI2018-098085-B-C44, Grant FC-GRUPIN-IDI/2018/000193, and under Grant FC-GRUPIN-IDI/2018/000226.

ABSTRACT A quantum procedure for testing the commutativity of a finite dimensional algebra is introduced. This algorithm, based on Grover's quantum search, is shown to provide a quadratic speed-up (when the number of queries to the algebra multiplication constants is considered) over any classical algorithm (both deterministic and randomized) with equal success rate and shown to be optimal among the class of probabilistic quantum query algorithms. This algorithm can also be readily adapted to test commutativity and hermiticity of square matrices, again with quadratic speed-up. The results of the experiments carried out on a quantum computer simulator and on one of IBM's 5-qubit quantum computers are presented.

INDEX TERMS Deterministic algorithms, randomized algorithms, quantum algorithms, commutativity, finite dimensional algebras.

I. INTRODUCTION

Quantum computation has been considered a promising technology since its introduction in the works of Feynman, Manin, Benioff and others [3], [10], [11], [27]. The theoretical potential of quantum computers to outperform classical computers has driven many researchers to introduce specific algorithms for this computational paradigm. Among the different ''animals'' in the quantum algorithm zoo [19], two ''species'' have received special attention from the scientific community because of their cryptographic consequences: Grover [14] and Shor [38] algorithms. Apart from them, not many classes of quantum algorithms have been discovered. In 2003, Shor pointed out that ''all the quantum algorithms known to offer substantial speed-up over classical algorithms for the same problems fall into one of three classes'' [39]. In the same paper, he tries to give explanations for this fact, and encourages some lines of research that might lead to the discovery of more quantum algorithms. In the last 15 years the situation has remained more or less the same, so some authors advocate now for widening the scope of problems in which the known quantum algorithms can be applied:

> ''As well as the development of new quantum algorithms, an important direction for future research seems to be the application of known quantum algorithms (and algorithmic primitives) to new problem areas. This is likely to require significant input from, and communication with, practitioners in other fields.'' [29]

In the meantime the actual technology has had a slow development. As of December of 2018 a handful of quantum computers with a small amount of qubits are available, notably those made publicly accessible on the cloud through IBM's Quantum Experience [16], and promises of computers with some dozens of qubits have been announced. This has spurred research on both the theoretical and the practical possibilities of quantum computing [13], [23], [28], [35].

In this context, we introduce our quantum algorithm for testing the commutativity of a finite dimensional algebra. It is based on Grover's algorithm (a method that has been successfully applied, for instance, in [4]) and it is intended to address the ''application of known quantum algorithms to new problem areas''. The problem of determining whether a given algebraic structure (algebra, ring, group,...) is commutative or not is traditional, and it has been theoretically studied in several contexts (e.g., in [18], [25], [32]). From an effective point of view, it has been specially considered in the case of groups, where algorithms (not only randomized classical but also quantum) are known [26], [31]. In the case of finite dimensional algebras, the need of an effective

The associate editor coordinating the review of this manuscript and approving it for publication was Dongxiao Yu.

procedure for testing commutativity is natural in the context of the computational study of finite semifields that we have carried out in recent years [6], [7], [33], [34]. In particular, our studies on four dimensional division rings over the finite field with seven elements required a processing of hundred of thousands of algebras in more than a million hours of computer time. Each one of those has $O(7^8)$ isotopes (for a definition of isotopic algebras, see Section II below), being the commutativity of one of them equivalent to the existence of a symplectic semifield plane of order 7^4 [20], leading to the interest of our algorithm.

The structure of the paper is as follows. In Section II we collect basic notions on finite dimensional algebras. Section III is devoted to classical algorithms (both deterministic and randomized) testing whether a finite dimensional algebra is commutative or not. In Section IV the main algorithm is presented together with its computational analysis. Section V is devoted to numerical simulations and to experiments carried out on one of IBM's 5-qubit quantum computers. Finally, in Section VI we draw some conclusions and present ideas for further research.

II. PRELIMINARIES

Let K be a field (it can be infinite such as the real or complex number fields, or finite, i.e., a Galois field $\mathbb{F}_q$ [24]). A $K-$algebra A is a $K-$vector space equipped with a bilinear product $\cdot$ [22]. It is called commutative (resp. associative) if the multiplication satisfies the commutativity (resp. associativity) property:

$$a \cdot b = b \cdot a \qquad (\text{resp. } (a \cdot b) \cdot c = a \cdot (b \cdot c))$$

for all $a, b, c \in A$. It is unital if the product has an identity element. Besides, it is known as division algebra if any nonzero element has left and right multiplicative inverses. A $K-$algebra is called finite-dimensional in case the underlying $K-$vector space is finite dimensional. Particular cases of $K-$algebras include matrix rings over the field K, Lie and Jordan algebras (i.e., $K-$algebras satisfying Lie or Jordan axioms [37]) or finite semifields (i.e., $\mathbb{F}_q-$finite dimensional division algebras [33]).

Given two elements $a, b \in A$, we define the commutator $[a, b] = ab - ba$. It can be straightforwardly checked that the sets $C(a) = \{b \in A \mid [a, b] = 0\}$ and $C(A) = \{a \in A \mid [a, b] = 0 \ \forall b \in A\} = \{a \in A \mid C(a) = A\}$ are $K-$vector subspaces of A. Also, it is clear that A is commutative if and only if $C(A) = A$.

If A is a $n-$dimensional $K-$algebra ($n \in \mathbb{N}$), and $B = \{x_1, \ldots, x_n\}$ is a $K-$basis of a A (i.e., $A = K < \{x_1, \ldots, x_n\} >$), then there exists a unique set of constants $\{M_{ijk}\}_{i,j,k=1}^n \subseteq K$ such that

$$x_i \cdot x_j = \sum_{k=1}^{n} M_{ijk} x_k \quad \forall i, j \in \{1, \ldots, n\}$$

This set of multiplication constants is also known as cubical array, 3-cube or multiplication table corresponding to A with respect to the basis B, and it completely determines the product in A. Notice that, for all $i = 1, \ldots, n$, the coordinate matrix of the $K-$linear map $L_{x_i} : A \to A$ given by $L_{x_i}(a) = x_i \cdot a$ is $(M_{ijk})_{k,j=1}^n$. Remark also that A is a commutative algebra if and only if $[x_i, x_j] = 0$, for all $i, j = 1, \ldots, n$, i.e., if and only if $M_{ijk} = M_{jik}$, for all $1 \leq i, j, k \leq n$.

As mentioned in the introduction, our motivation for the study of the commutativity of finite-dimensional algebras comes from its importance in the classification of semifields (which are a particular case of algebras), where many tests of commutativity must be performed. For this reason, in this paper we focus on finding a speed-up for the decision problem "Given a finite dimensional $K-$algebra A, is A commutative?" by using quantum computing.

III. CLASSICAL ALGORITHMS FOR TESTING THE COMMUTATIVITY OF A FINITE DIMENSIONAL ALGEBRA

To the best of our knowledge, the problem of determining with a classical algorithm whether a given $K-$algebra is commutative or not has not been explicitly studied in the literature. However, in this section we collect classical (randomized or not) algorithms for that problem and we study the number of queries needed in the general case. We will consider a finite dimensional $K-$algebra A with basis $B = \{x_1, \ldots, x_n\}$. In this setting, it seems natural to consider as input data the multiplication constants of A with respect to B. They consist on n^3 elements in the field K. Access to the input data will be query modeled, i.e., we will assume there is an oracle providing the multiplication constants of the algebra, on demand. Namely, any access to one of these multiplication constants will be counted as one query. Any other operation involved in the algorithms is considered query free.

A first attempt to determine the commutativity of A is to translate, into the language of algebras, the probabilistic algorithm for testing group commutativity presented in [31]. It is therefore based on a direct computation of commutators of A.

Algorithm 1 Computation of Commutators of Random Elements of A

```
Fix t ∈ ℕ
For i from 1 to t do:
    Pick uniformly (and independently) elements a, b ∈ A
    Compute the commutator [a, b]
    If [a, b] ≠ 0, return NO
Return YES
```

Proposition 1: Algorithm 1 requires $O(tn^3)$ queries. On output NO it provides an accurate answer, while on output YES the error probability is bounded.

Proof: The algorithm performs, at most, $2t$ multiplications in A, hence the number of algebra queries (observe that each product requires access to n^3 multiplication constants).

Let C be a proper $m-$dimensional $K-$vector subspace of A. If a is picked uniformly on A, then $P(a \notin C) = 1$ if K is

infinite, where as $P(a \notin C) = 1 - P(a \in C) = 1 - \frac{q^m}{q^n} = \frac{q^{n-m}-1}{q^{n-m}}$, when $K = \mathbb{F}_q$ is finite. For the algorithm to produce a wrong answer, it is necessary that A is not commutative while all the picked random pairs $(a, b) \in A^2$ commute. When A is not commutative, $C(A)$ is a proper $K-$vector subspace of A of dimension at most $n-2$ (this is because $\dim_K C(A) = n-1$ allows us to write any element in the algebra as $c+\lambda d$, where $c \in C(A)$, $\lambda \in K$ and $d \notin C(A)$ is a prefixed element, so that $[c + \lambda d, c' + \lambda' d] = [c, c' + \lambda' d] + [\lambda d, c'] + \lambda\lambda'[d, d] = 0$, i.e., A is commutative, a contradiction). Moreover, if $a \notin C(A)$, then $C(a)$ is also a proper $K-$vector subspace of A. So, the probability that a random pair does not commute is

$$\begin{aligned} P([a, b] \neq 0) &= P(a \notin C(A) \ \wedge \ b \notin C(a)) \\ &= P(a \notin C(A)) \cdot P(b \notin C(a) \ / \ a \notin C(A)) \\ &\geq \left(\frac{q^2-1}{q^2}\right) \cdot \left(\frac{q-1}{q}\right) \end{aligned} \quad (1)$$

if $K = \mathbb{F}_q$ is finite, and equal to 1, when K is infinite (incidentally, observe that this probability agrees with the limit of previous bound when taking q to infinity). In both cases it is a constant probability independent of n. ■

The main inconvenient to this approach is the amount of queries required, as it is in the order of the number of queries for the straightforward algorithm consisting in checking pairs of corresponding multiplication constants. Namely,

Algorithm 2 Exhaustive Multiplication Constant Testing

```
For k from 1 to n do:
    For i from 1 to n do:
        For j from i + 1 to n do:
            If M_ijk ≠ M_jik, return NO
Return YES
```

Proposition 2: Algorithm 2 always gives the right answer requiring $O(n^3)$ queries.

Proof: The number of queries is $n \cdot n \cdot \frac{n-1}{2} \cdot 2 = n^3 - n^2$, which is exactly the number of constants M_{ijk} with $i \neq j$. ■

A compromise between both approaches is the randomization of Algorithm 2 using the ideas of Algorithm 1.

Algorithm 3 Randomized Multiplication Constant Testing

```
Fix t ∈ ℕ
For i from 1 to t do:
    Pick uniformly (and independently) integers i, k ∈ {1, ..., n}
    Pick uniformly an integer j ∈ {1, ..., n} \ {i}
    If M_ijk ≠ M_jik, return NO
Return YES
```

Proposition 3: Algorithm 3 requires $O(t)$ queries. On output NO it provides an accurate answer, while on output YES the error probability is at most $\left(\frac{n^3-n^2-2}{n^3-n^2}\right)^t$.

Proof: The algorithm requires, at most, $2t$ multiplication constant queries. On the other hand, for a noncommutative finite dimensional $K-$algebra A there must exist $i, k \in \{1, \ldots, n\}, j \in \{i+1, \ldots, n\}$ such that $M_{ijk} \neq M_{jik}$. Therefore, the probability of wrongly declaring A as commutative is at most $\left(\frac{\frac{n^3-n^2}{2}-1}{\frac{n^3-n^2}{2}}\right)^t$. ■

Remark 1: The error probability of Algorithm 3 depends of the dimension of the algebra, and approaches 1 as we make n bigger. This marks a difference with the bounded error probability of Algorithm 1. The price to pay, of course, is the number of oracle queries required for the execution of this algorithm.

The main drawback of this approach is that the error probability can not be improved in general. This is due to the fact that one different pair of corresponding multiplication constants suffices for the algebra to be noncommutative. This fact yields the following result.

Proposition 4: If $f(n)$ is a function such that $\lim_{n\to\infty} \frac{f(n)}{n^3} = 0$ and M is an algorithm for the problem of testing commutativity of finite-dimensional $K-$algebras with the three following properties

1) *For algebras of dimension n, M uses at most $f(n)$ queries to the multiplication constants*
2) *There exists $a \geq 0$ such that for any commutative $K-$algebra A it holds $Pr(YES|A) \geq a$*
3) *There exists $b \geq 0$ such that for any non-commutative $K-$algebra A it holds $Pr(NO|A) \geq b$*

then $a + b \leq 1$.

Proof: This kind of result could be proved by means of Yao's minimax principle [40] (cf., for instance, [15] page 60 for the classical case related to Grover's search algorithm), but here we will opt for a more direct, constructive approach.

We fix a dimension n and define A_{ijk} to be the $K-$algebra whose multiplication constants verify $M_{ijk} = 1$ and $M_{stu} = 0$ whenever $(i, j, k) \neq (s, t, u)$. Consider the set $\mathcal{A}_n = \{A_{ijk} : i \neq j\}$. Notice that every $A \in \mathcal{A}_n$ is non-commutative and it has exactly two constants such that $M_{ijk} \neq M_{jik}$. We are going to estimate $Pr(YES|A \in \mathcal{A}_n)$, that is, the probability that the algorithm says that A is commutative when we take A from $\mathcal{A}_n$.

For that, we first estimate the probability that, given that the algorithm examines only the positions contained in a certain set P of triples and that A is taken from $\mathcal{A}_n$, the unique position (i, j, k) on which $M_{ijk} \neq 0$ for A is not contained in P. If $|P| = p$, this happens with probability at least

$$\frac{n^3 - n^2 - p}{n^3 - n^2} \geq \frac{n^3 - n^2 - f(n)}{n^3 - n^2} =: h(n)$$

because the constant $M_{ijk} = 1$ can be any of the triples (i, j, k) with $i \neq j$, which are $n^3 - n^2$, and the "favorable" cases are those in which the position is not in P, so we need to exclude the p positions queried by the algorithm. Notice that $h(n)$ tends to 1 when n tends to infinity since $\lim_{n\to\infty} \frac{f(n)}{n^3} = 0$. Also, the lower bound holds for any P but is independent of P, so

$$Pr(\text{witness undetected}|A \in \mathcal{A}_n) \geq h(n)$$

Now, if the algorithm doesn't find the unique witness of non-commutativity, then the execution will be exactly the same as if the input were a commutative algebra and, by hypothesis, the answer will be YES with probability at least a. That is

$$Pr(YES|A \in \mathcal{A}_n, \text{witness undetected}) = Pr(YES|A \text{ is commutative})$$

Thus, abbreviating "witness undetected" by "wu", we have

$$Pr(YES|A \in \mathcal{A}_n) \geq Pr(\text{wu}|A \in \mathcal{A}_n)Pr(YES|A \in \mathcal{A}_n, \text{wu}) \geq ah(n)$$

since it is easy to verify that $Pr(x|y) \geq Pr(z|y)Pr(x|y, z)$.

Consequently

$$Pr(NO|A \in \mathcal{A}_n) \leq 1 - ah(n)$$

We also know that $Pr(NO|A) \geq b$ for all non-commutative A, so $Pr(NO|A \in \mathcal{A}_n) \geq b$ and, consequently, we have $b \leq Pr(NO|A \in \mathcal{A}_n) \leq 1 - ah(n)$, which implies

$$b + ah(n) \leq 1$$

Taking limits on both sides of the inequality we obtain the result. ∎

In this context a natural question arises: is it possible that for some specific class of noncommutative algebras the number of "failing" pairs (i.e., noncommutativity witnesses) of corresponding constants must be greater than one? In particular, what can be said for finite semifields? In this case, since the algebra is unital, its unit commutes with any other element, so the number of different pairs of constants is at most $\frac{(n-2)(n-1)}{2} \cdot n = \frac{n^3-3n^2+2n}{2}$. We have studied this problem for finite semifields of small orders (as complete classifications are known), finding the existence of noncommutative finite semifields of orders 2^4 and 3^3 with only one "failing" pair, while others of orders 3^4 and 3^5 with maximal number of noncommutativity witnesses (12 and 30, respectively).

In view of the analysis of the randomized classical algorithms presented in this section, we think that an approach based on quantum computations seems reasonable. Our hope is that the power of quantum computation improves the previous algorithms, specially from the point of view of the number of queries required to test the commutativity of the algebra without sacrificing the constant (on n) error probability. We consider this approach in the next section.

IV. A QUANTUM ALGORITHM FOR TESTING THE COMMUTATIVITY OF A FINITE DIMENSIONAL ALGEBRA

A quantum algorithm for deciding the commutativity of a finite dimensional $K-$algebra A with basis $B = \{x_1, \ldots, x_n\}$ is introduced below. As in the previous section the multiplication constants of A with respect to B will be the query modeled input data. This implies that a direct translation of the quantum methods proposed in [26] for testing the commutativity of a group based of commutators of elements, such as we did in Algorithm 1, is useless (just notice that the computation of a single product requires $\Omega(n^3)$ oracle queries).

We will further assume that the multiplication constants, which belong to the field K, are given by an $l-$bit representation. For instance, if $K = \mathbb{F}_q$ is the finite field with q elements, then l can be taken as $\lceil \log_2 q \rceil$. On the other hand, if K is an infinite field (such as $\mathbb{R}$ or $\mathbb{C}$), then an standard numerical representation can be used. In this case, the potential error derived from the inaccuracy of this representation has to be taken into account. Let us say by now that our algorithm, not making any arithmetic operation with the constants, will not introduce new numerical errors.

In order to execute the algorithm on an actual computer, it is convenient that the dimension n is a power of two. Also, in the proof of Theorem 1 we will need a bound on the number of triples (i, j, k) such that $M_{ijk} \neq M_{jik}$. For these reasons, we will embed our algebra A in an algebra $\hat{A}$ with holds the same commutative character as A.

Lemma 1: For any $n \in \mathbb{N}$, take $m \in \mathbb{N}$ such that $2^{m-1} \leq \sqrt[3]{\frac{4}{3}(n^3 - n^2)} < 2^m$. Then, $\hat{n} = 2^m \leq \sqrt[3]{\frac{32}{3}}n$, and the $\hat{n}-$dimensional algebra $K-$algebra $\hat{A} = A \times K^{\hat{n}-n}$ with the product given by the rule $(a, \lambda) \cdot (b, \mu) = (ab, 0)$, is commutative if and only A is. Moreover, the number of multiplicative constants of $\hat{A}$ such that $\hat{M}_{ijk} \neq \hat{M}_{jik}$ is less than $\frac{3}{4}\hat{n}^3$.

Proof: It is clear that $\hat{n} = 2 \cdot 2^{m-1} \leq 2 \cdot \sqrt[3]{\frac{4}{3}(n^3 - n^2)} \leq \sqrt[3]{\frac{32}{3}}n$. Also, the set $\hat{B} = \{(x_i, 0)\}_{i=1}^{n} \cup \{(0, e_i)\}_{i=1}^{\hat{n}-n}$ is a $K-$basis of $\hat{A}$ ($\{e_i\}_{i=1}^{n}$ is the standard basis of $K^{\hat{n}-n}$). The corresponding multiplication constants are $\hat{M}_{ijk} = M_{ijk}$ if $1 \leq i, j, k \leq n$, and $\hat{M}_{ijk} = 0$ otherwise. Therefore A and $\hat{A}$ are simultaneously commutative or not. Finally, the number of multiplicative constants such that $\hat{M}_{ijk} \neq \hat{M}_{jik}$ is at most (see proof of Proposition 2) $n^3 - n^2 < \frac{3}{4}\hat{n}^3$. ∎

Remark 2: From the proof of the previous lemma it is clear that $\hat{n} = \Theta(n)$. Notice also that, if we are given a query oracle O^A for the multiplication constants of A, then a second oracle $O^{\hat{A}}$ for the multiplication constants of $\hat{A}$ can be easily made. It should return the output of the $A-$oracle if $1 \leq i, j, k \leq n$, and 0 otherwise. So, the $\hat{A}-$oracle only requires one access to the $A-$oracle, at most.

Now, we can introduce a quantum algorithm for testing the commutativity of a finite-dimensional $K-$algebra A. Our algorithm will call Grover's quantum search algorithm, as described in [30, Chapter 6.1]. In this setting we model the oracle $O^{\hat{A}}$ in such a way that $3m$ index register qubits provide the encoding of the triple ijk while the multiplication constant $\hat{M}_{ijk}$ is added to the l oracle qubits (see figure 1).

Calls to the quantum search algorithm require an specific oracle made from the oracle $O^{\hat{A}}$. Namely, the function f for which solutions are searched is (suitably encoded) $f(ijk) = 1 - \delta_{\hat{M}_{ijk}, \hat{M}_{jik}}$, i.e., $f(ijk) = 1$ if and only if $\hat{M}_{ijk} \neq \hat{M}_{jik}$. This function can be straightforwardly modeled as an oracle O^f requiring a constant number of queries to the oracle $O^{\hat{A}}$. The

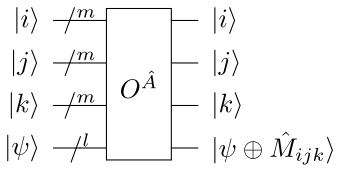

FIGURE 1. Multiplication constant oracle.

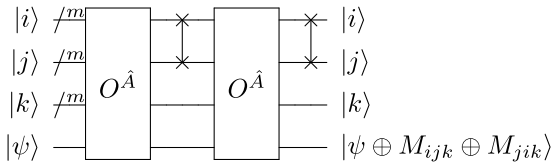

FIGURE 2. Grover oracle O^f when $K = \mathbb{F}_2$.

actual number of queries and the form of this oracle depends on the field K. When K is the binary field, the Grover oracle is the one in figure 2.

Observe that in this situation an advantage is taken from the matching of the field addition and the mod 2 addition of the computational basis qubits, and so only two oracle calls are needed (and two swap operators for the interchanging of the indexes i and j). However, for any other field K further operations are required. First, the evaluation of the equality of the multiplication constants $\hat{M}_{ijk}$ and $\hat{M}_{jik}$, which is realized by a combination of NOT and $n-$Toffoli gates. Secondly, the restoring of the qubits (including the l ancilla qubits) to their original state, so that they can be used in the next iterations of Grover's algorithm (see figure 3).

As we can see, four $O^{\hat{A}}$ queries are required in this case. Nevertheless, for any field K (equal to $\mathbb{F}_2$ or not), the number of oracle queries to $O^{\hat{A}}$ is in the same order of the number oracle queries to O^f.

Next, the algorithm is finally presented together with the analysis of its oracle query optimality among quantum algorithms.

Main algorithm Quantum Multiplication Constant Testing

Pick uniformly an integer $l \in \left\{0, \ldots, \sqrt{\frac{\hat{n}^3}{2}} - 1\right\}$
Initialize the state to $\sum_{x=1}^{\hat{n}^3} \frac{1}{\sqrt{\hat{n}^3}}|x\rangle$
Apply l iterations of Grover quantum search
Observe to estimate ijk
Call the oracle $O^{\hat{A}}$ with ijk and jik
If $\hat{M}_{ijk} \neq \hat{M}_{jik}$, return NO
Else, return YES

Theorem 1: The main algorithm requires $\Theta(\sqrt{n^3})$ queries to the oracle O^A. On output NO it always provides an accurate answer, while on output YES the error probability is a constant strictly less than 1. Moreover, it is query-optimal among the quantum algorithms, in the sense that any other algorithm with bounded error probability for testing the commutativity of the finite dimensional $K-$algebra A, uses $\Omega(\sqrt{n^3})$ queries.

Proof: Notice first that the main algorithm requires at most $4\sqrt{\frac{\hat{n}^3}{2}} - 2 = \Theta(\sqrt{n^3})$ queries to the oracle O^A (this can be clearly seen from Remark 2 and the comments below it). Also, the algorithm is accurate when the answer is NO (as it directly tests the existence of a pair of multiplication constants witnessing the noncommutativity of A). On the other hand, on answer YES the probability of error is derived from the iterations of the Grover quantum search algorithm. If we assume that A is actually a noncommutative $K-$algebra, then the number of multiplication constants M_{ijk} differing from the corresponding M_{jik} is $t \geq 2$. If $0 < \theta < \frac{\pi}{2}$ is such that $\sin^2\theta = \frac{t}{\hat{n}^3}$, then $\frac{1}{\sin 2\theta} = \frac{\hat{n}^3}{2\sqrt{t(\hat{n}^3 - t)}}$. Lemma 1 gives us $t < \frac{3}{4}\hat{n}^3$, and so $\frac{1}{\sin 2\theta} < \sqrt{\frac{\hat{n}^3}{t}} \leq \sqrt{\frac{\hat{n}^3}{2}}$. Therefore, we can apply [5, Lemma 2] to get that the probability of the algorithm not finding a witness pair for the noncommutativity of A is at most $\frac{3}{4}$, and so constant. This finishes the analysis of the main algorithm.

Optimality of the algorithm can be directly derived from [1, Theorem 5.1]. Namely, consider the Grover function f. Take the set X consisting on the zero algebra, and the set Y whose elements are the $n^3 - n^2$ noncommutative algebras whose multiplication constants are all but one equal to zero (i.e., $M_{ijk} = 1$ for some $1 \leq i, j, k \leq n$ with $i \neq j$, and zero otherwise). Taking $R = X \times Y$ it is clear that for every $x \in X$, there exist at least $n^3 - n^2$ different $y \in Y$ such that $(x, y) \in R$, that for every $y \in Y$, there exists at least 1 different $y \in Y$ such that $(x, y) \in R$, that for every $x \in X$ and $1 \leq i, j, k \leq n$, there is at most 1 different $y \in Y$ such that $(x, y) \in R$ with $M^x_{ijk} \neq M^y_{ijk}$, and that for every $y \in Y$, and $1 \leq i, j, k \leq n$, there is at most 1 different $x \in X$ such that $(x, y) \in R$ with $M^x_{ijk} \neq M^y_{ijk}$. So, [1, Theorem 5.1] yields the desired lower bound of $\Omega(n^3)$ oracle queries. ∎

Remark 3: In virtue of Proposition 4, any classical algorithm with the same success probability as our quantum algorithm will require $\Omega(n^3)$ queries to the multiplication constants.

Remark 4: Observe that the extremal case $l = 0$ corresponds to randomly choosing a constant M_{ijk} and testing whether $M_{ijk} = M_{jik}$ or not (only two oracles queries are needed in this case).

Remark 5: If the Grover oracles are subtly adapted, then the previous algorithm can also be used to test whether a matrix is symmetric or hermitian. Simply use two index qubits instead of three and adopt a suitable representation of the complex field in the later case. Obviously, in this situation the number of queries would be $\Theta(n)$.

V. NUMERICAL EXPERIMENTS AND IMPLEMENTATION ON QUANTUM HARDWARE

To test the actual performance of the quantum algorithm, we have implemented an (exact) simulation of its behavior

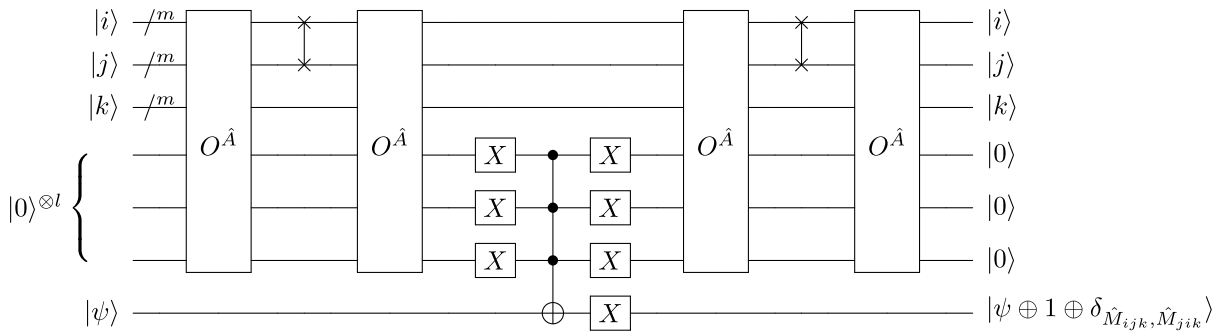

FIGURE 3. Grover oracle O^f when $K \neq \mathbb{F}_2$.

TABLE 1. Number of queries and success probabilities of the classical and quantum algorithms.

Dimension	Queries	Classic Alg. 2	Classic Alg. 3	Quantum Alg.
2	6	1	0.875	0.625
3	22	1	0.7263	0.608
4	62	1	0.7327	0.595
5	62	0.62	0.4654	0.595
6	62	0.3444	0.2928	0.595
7	62	0.2109	0.1907	0.595
8	182	0.4063	0.3345	0.5982
9	182	0.2809	0.2452	0.5982
10	182	0.2022	0.1833	0.5982
11	182	0.1504	0.1398	0.5982
12	182	0.1149	0.1086	0.5982
13	182	0.0897	0.0859	0.5982
14	182	0.0714	0.0690	0.5982
15	510	0.1619	0.1495	0.5946
16	510	0.1328	0.1244	0.5946
17	510	0.1103	0.1045	0.5946
18	510	0.0926	0.0885	0.5946
19	510	0.0785	0.0755	0.5946
20	510	0.0671	0.0649	0.5946
21	510	0.0578	0.0562	0.5946
22	510	0.0502	0.0489	0.5946
23	510	0.0438	0.0429	0.5946
24	510	0.0385	0.0378	0.5946
25	510	0.0340	0.0334	0.5946
26	510	0.0302	0.0297	0.5946
27	510	0.0269	0.0265	0.5946
28	510	0.0241	0.0238	0.5946
29	510	0.0217	0.0214	0.5946

in Matlab and we have run the most challenging situation (namely, when there is only a pair of values (i, j) and only a k such that $M_{ijk} \neq M_{jik}$) for algebras of dimensions from 2 to 29. Notice that, because of the symmetry of Grover's algorithm, the probability of success (that is, of finding a witness for non-commutativity) is equal for all the algebras of the same dimension and with the same number of non-commuting elements (and it is independent of the size of the underlying field). We have also compared this probability of success with the one obtained with the classical algorithms presented in Section III when implemented in a classical computer. To make a fair comparison, we allow the classical algorithms to consult the multiplication constants as many times as the quantum algorithm does in the worst case, namely $4\sqrt{\frac{\hat{n}^3}{2}} - 2$ (cf. the proof of Theorem 1). Notice, though, that in average the quantum algorithm will make only half that number of queries and that if the underlying field is $\mathbb{F}_2$ then the number of queries is, again, halved (see Figures 2 and 3). Since the number of queries is less than n^3, this excludes Algorithm 1 from the comparison, for it needs at least n^3 queries to complete an iteration. We also adjust Algorithm 2 to run for only a fixed number of queries and

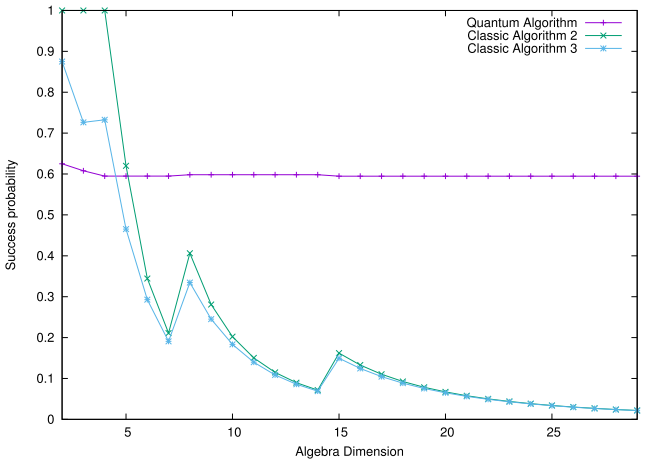

FIGURE 4. Success probability of quantum versus classical algorithms.

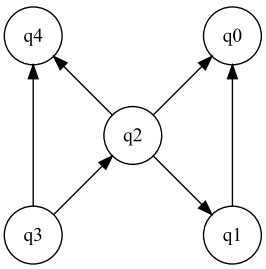

FIGURE 5. Connectivity of qubits on the ibqmx4.

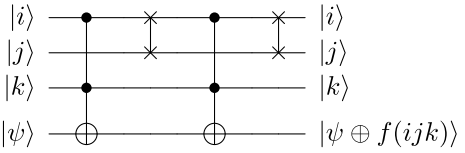

FIGURE 6. Experiment O^f.

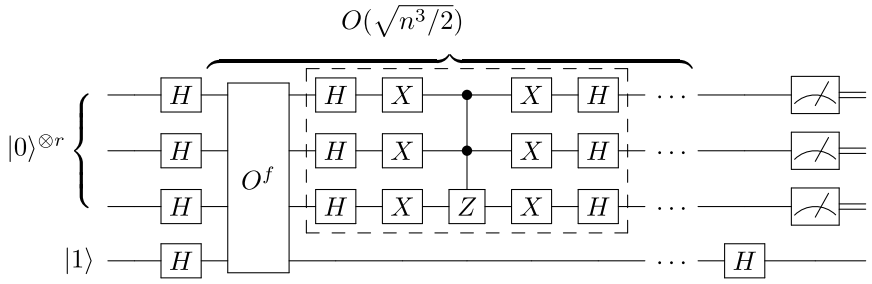

FIGURE 7. Grover algorithm without ancilla bits.

then stop, returning *YES* if it has not been able to find a pair of non-commuting elements. Its success probability will then be $\frac{tk}{n^3-n^2}$, where k is the number of non-commuting constants in the algebra and t is the number of iterations. In the case of Algorithm 3, this probability is $1-(\frac{n^3-n^2-k}{n^3-n^2})^t$. Note that both Algorithm 2 and Algorithm 3 require two queries per iteration, since they need to compare M_{ijk} and M_{jik}.

In Table 1 and in Figure 4, we present the results of these experiments. Notice that, following Lemma 1, in the quantum algorithm we sometimes need to embed the algebra in another with a higher dimension. This causes the number of queries and the success probability of the quantum algorithm to be the same for all dimensions augmented to the same 2^m and it explains the sudden increase in success probability of the classical algorithms for those dimensions where the embedding jumps from dimension 2^m to 2^{m+1}. Note that the quantum algorithm beats both classical algorithms for dimensions greater than 5 and, in fact, the success probability of the classical algorithms tends to zero when the dimension tends to infinity (cf. Proposition 4). Note also that the success probability of the quantum algorithm remains almost constant, as expected from Theorem 1. In fact, in this case the probability is about $\frac{3}{5}$, greater than the value guaranteed by the theorem, which was $\frac{1}{4}$.

As a proof of concept of a possible implementation on an actual quantum computer when fault-tolerant quantum hardware is available, the main algorithm has been implemented (after successfully been tested on a quantum computer simulator) on ibmqx4, one of the IBM 5-qubit computers publicly accessible through IBM's Quantum Experience cloud services ([16]). The processor of this computer uses 5 superconducting transmon qubits. Available 1-qubit gates include X, Y, Z, H, S, $S^\dagger$, T, $T^\dagger$ as well as CNOT gates between some pairs of qubits, with controls and targets as depicted in figure 5. More details on the architecture can be found on [17].

The number of qubits of the quantum computer limits the size of the field and the matrices that can be considered. The design of the oracle O^f forces the election of $m = l = 1$, so A must be a $2-$dimensional algebra over the binary field. We have opted for the following product on $A = \mathbb{F}_2 \times \mathbb{F}_2$:

$$(a, b) \cdot (c, d) = (0, ad + bd)$$

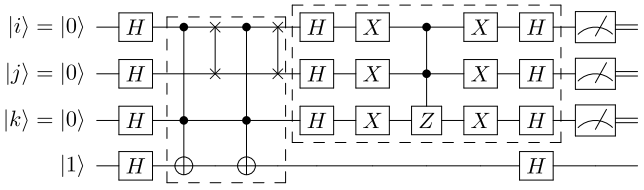

FIGURE 8. Experiment.

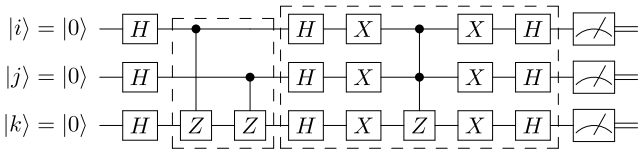

FIGURE 9. Experiment with phase oracle.

Observe that A is a noncommutative algebra, since $(0, 1) \cdot (1, 0) \neq (1, 0) \cdot (0, 1)$. Its multiplication constants M_{ijk} are:

$$k = 1, 2 \left\{ \overbrace{\overbrace{\begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}}^{j=1,2} \overbrace{\begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}}^{j=1,2}}^{i=1,2} \right.$$

The indexes ijk of the multiplication constants, in the range $\{1, 2\}$, will be encoded as qubits $|0\rangle$ and $|1\rangle$. Using this representation the oracle O^A is easily seen to be equal to $|j \text{ AND } k\rangle$, which will greatly simplify its wiring (in particular, no ancilla bits will be used). Lemma 1 gives us $\hat{n} = 2 = n$, and so there is no need for the embedding of A into $\hat{A}$. The Grover oracle in our particular case is the one in figure 6. where $f(ijk) = (i \text{ AND } k) \oplus (j \text{ AND } k)$. Among the 8 different multiplication constants M_{ijk} only 2 (exactly one fourth) differ from the corresponding M_{jik}, so we are in the particular nice case described in [5, Section 3.1]. When Grover quantum search is applied (i.e., when $l \neq 0$), only one iteration is required, providing a certain answer with two O^A queries. So, the general Grover circuit (figure 7) specializes in our particular instance as that of figure 8.

Further simplifications were achieved from the use of a phase oracle (see, for instance [12]). Namely, the state of the qubits in the previous experiment just before applying the oracle is $(\sum_{x=1}^{\hat{n}^3} \frac{1}{\sqrt{\hat{n}^3}}(-1)^{f(x)}|x\rangle) \otimes H|1\rangle$. As the measurement

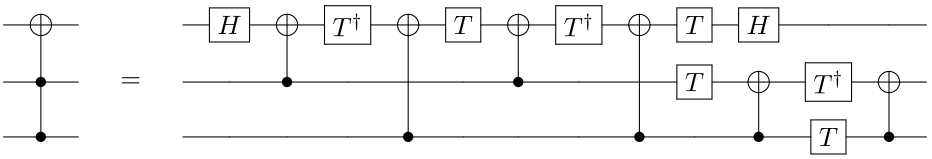

FIGURE 10. Toffoli gate in the *ibmqx4*.

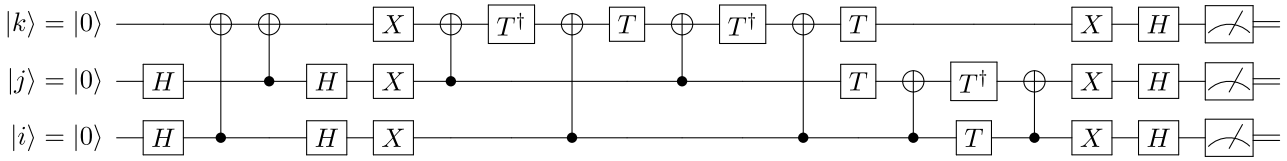

FIGURE 11. Experiment with phase oracle in the *ibmqx4*.

TABLE 2. Results of the experiment on the actual quantum computer.

State	Observations
$\|000\rangle$	28
$\|001\rangle$	123
$\|010\rangle$	52
$\|011\rangle$	299
$\|100\rangle$	63
$\|101\rangle$	356
$\|110\rangle$	49
$\|111\rangle$	54

is only taken in the first qubits, the relevant part of the state is $\sum_{x=1}^{\hat{n}^3} \frac{1}{\sqrt{\hat{n}^3}}(-1)^{f(x)}|x\rangle$. We can realize the transformation $|x\rangle \to \sum_{x=1}^{\hat{n}^3} \frac{1}{\hat{n}^3}(-1)^{f(x)}|x\rangle$ without the need of the last qubit, simply through the use of the phase oracle in figure 9.

Finally, because of the specific architecture of the *ibmqx4* quantum computer we need to use the (optimal, see [36]) gate substitution of figure 10.

Summarizing, the actual circuit to be tested is presented in figure 11 (for convenience, a reordering of the qubits have been made).

As expected, the outcome of the experiment carried out in the quantum computer simulator provided by the IBM Quantum Experience gives exact results, with only two states of equal nonzero probability, $|011\rangle$ and $|101\rangle$. The performance of the quantum computer ibmqx4, due to the intrinsic error of the implementation of the quantum gates, makes the outcome of the experiment a little bit fuzzier, as can be seen on Table 2. Out of a total of 1024 runs of the experiment, 299 (29.19%) resulted on an observation of the state $|011\rangle$ and 356 (34.76%) resulted on $|101\rangle$. In the remaining 369 cases (36.03%), other, non successful states were obtained due to the accumulated errors of the gates of the circuit. Note that this execution on quantum hardware corresponds to the case when we obtain $l = 1$ in the first step of the quantum algorithm. Since in this case $\hat{n} = 2$, the other possibility is $l = 0$ (that is, we select elements ijk uniformly at random, cf. Remark 4) that has success rate 0.25. To obtain the overall success probability of the algorithm for this algebra, we need to average both situations, getting 0.4448. The difference with the 0.625 obtained in the exact simulations (see the first row of Table 1) comes from the noise of current quantum hardware and will be overcome once fault-tolerant quantum computing devices are available.

VI. CONCLUSIONS AND FUTURE WORK

In this paper we have studied the problem of effectively determining when a finite-dimensional algebra A over a field is commutative. Any classical algorithm (randomized or not) that solves the problem with bounded error needs $\Omega(n^3)$ queries to the multiplication constants of the algebra, where n is the dimension of A. However, we have introduced a quantum algorithm based on Grover's search method that uses $O(\sqrt{n^3})$ queries and has no error on a NO answer and bounded error on a YES answer. We have also shown that this method is query-optimal among all quantum algorithms and that it can be implemented with $O(\log(n))$ qubits. This algorithm can also be readily adapted to test commutativity and hermiticity of $n \times n$ matrices with just $O(n)$ queries. We have also successfully tested our main algorithm with numerical simulations for algebras of dimension ranging from 2 to 29 and for a simple case of a $2-$dimensional algebra over $\mathbb{F}_2$ on an actual quantum computer, namely the imbqx4 publicly accessible through the IBM Quantum Experience.

On the light of these results, the application of quantum computing to problems related to the one studied in this paper seems very promising as does the use of other techniques, such as quantum walks [2], [8], [21] and adiabatic computing [9], to the same problem. In future works, we would like to approach the design and implementation of quantum algorithms for problems such as the determination of isomorphism and isotopy between semifields and the possibility of applying quantum computing to speed-up tasks such that the classification of all finite semifields of size 128, which is completely out of reach with current, classical computing technology.

REFERENCES

[1] A. Ambainis, "Quantum lower bounds by quantum arguments," *J. Comput. Syst. Sci.*, vol. 64, no. 4, pp. 750–767, 2002.

[2] A. Ambainis, "Quantum walks and their algorithmic applications," *Int. J. Quantum Inf.*, vol. 1, no. 4, pp. 507–518, 2003.

[3] P. Benioff, "The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines," *J. Stat. Phys.*, vol. 22, no. 5, pp. 563–591, May 1980.

[4] P. Botsinis, Y. Huo, D. Alanis, Z. Babar, S. X. Ng, and L. Hanzo, "Quantum search-aided multi-user detection of IDMA-assisted multi-layered video streaming," *IEEE Access*, vol. 5, pp. 23233–23255, 2017.

[5] M. Boyer, G. Brassard, P. Høyer, and A. Tapp, "Tight bounds on quantum searching," *Prog. Phys.*, vol. 46, nos. 4–5, pp. 493–505, 1998.

[6] E. F. Combarro, I. F. Rúa, and J. Ranilla, "New advances in the computational exploration of semifields," *Int. J. Comput. Math.*, vol. 88, no. 9, pp. 1990–2000, 2011.

[7] E. F. Combarro, I. F. Rúa, and J. Ranilla, "Finite semifields with 7^4 elements," *Int. J. Comput. Math.*, vol. 89, nos. 13–14, pp. 1865–1878, 2012.

[8] E. F. Combarro, J. Ranilla, and I. F. Rúa, "Quantum walks for the determination of commutativity of finite dimensional algebras," *J. Comput. Appl. Math.*, vol. 354, pp. 496–506, Jul. 2019.

[9] E. F. Combarro, J. Ranilla, and I. F. Rúa, "Experiments testing the commutativity of finite-dimensional algebras with a quantum adiabatic algorithm," *Comput. Math. Methods*, vol. 1, no. 1, 2019, Art. no. e1009.

[10] D. Deutsch, "Quantum theory, the Church–Turing principle and the universal quantum computer," *Proc. Roy. Soc. London A, Math. Phys. Sci.*, vol. 400, pp. 97–117, Jul. 1985.

[11] R. P. Feynman, "Simulating physics with computers," *Int. J. Theor. Phys.*, vol. 21, nos. 6–7, pp. 467–488, 1982.

[12] C. Figgatt, D. Maslov, K. A. Landsman, N. M. Linke, S. Debnath, and C. Monroe, "Complete 3-Qubit Grover search on a programmable quantum computer," *Nature Commun.*, vol. 8, no. 1, 2017, Art. no. 1918.

[13] D. Ghosh, P. Agarwal, P. Pandey, B. K. Behera, and P. K. Panigrahi, "Automated error correction in IBM quantum computer and explicit generalization," *Quantum Inf. Process.*, vol. 17, no. 6, p. 153, 2018.

[14] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proc. 28th Annu. ACM Symp. Theory Comput.*, 1996, p. 212.

[15] M. Hayashi, S. Ishizaki, A. Kawachi, G. Kimura, and T. Ogawa, *Introduction to Quantum Information Science*. Berlin, Germany: Springer, 2015.

[16] *IBM Quantum Experience*. Accessed: Apr. 5, 2019. [Online]. Available: https://www.research.ibm.com/ibm-q/

[17] *IBM QX4: Raven*. Accessed: Apr. 5, 2019. [Online]. Available: https://ibm.biz/qiskit-ibmqx4

[18] N. Jacobson, "Structure theory for algebraic algebras of bounded degree," *Ann. Math.*, vol. 46, pp. 695–707, Oct. 1945.

[19] S. Jordan. (17, 2011). *Quantum Algorithm Zoo*. [Online]. Available: http://math.nist.gov/quantum/zoo/

[20] W. M. Kantor, "Commutative semifields and symplectic spreads," *J. Algebra*, vol. 270, pp. 96–114, Dec. 2003.

[21] J. Kempe, "Quantum random walks: An introductory overview," *Contemp. Phys.*, vol. 44, pp. 307–327, Jul. 2003.

[22] S. Lang, *Algebra*. Reading, MA, USA: Addison-Wesley, 1965.

[23] H.-S. Li, X. Chen, H. Xia, Y. Liang, and Z. Zhou, "A quantum image representation based on bitplanes," *IEEE Access*, vol. 6, pp. 62396–62404, 2018.

[24] R. Lidl and H. Niederreiter, *Finite Fields* (Encyclopedia of Mathematics and its Applications), vol. 20. Reading, MA, USA: Addison-Wesley, 1983.

[25] J. H. Maclagan-Wedderburn, "A theorem on finite algebras," *Trans. Amer. Math. Soc.*, vol. 6, pp. 349–352, Jul. 1905.

[26] F. Magniez and A. Nayak, "Quantum complexity of testing group commutativity," in *Automata, Languages and Programming* (Lecture Notes in Computer Science). vol. 3580. Berlin, Germany: Springer, 2005.

[27] Y. Manin, *Vychislimoe I Nevychislimoe*. Moscow, Russia: SOV Radio, 1980, pp. 13–15.

[28] F.-X. Meng, X.-T. Yu, R.-Q. Xiang, and Z.-C. Zhang, "Quantum algorithm for spectral regression for regularized subspace learning," *IEEE Access*, vol. 7, pp. 4825–4832, 2019.

[29] A. Montanaro, "Quantum algorithms: An overview," *npj Quantum Inf.*, vol. 2, Jun. 2016, Art. no. , Art. no. 15023. doi: 10.1038/npjqi.2015.23.

[30] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, 10th ed. Cambridge, U.K.: Cambridge Univ. Press, 2011.

[31] I. Pak, "Testing commutativity of a group and the power of randomization," *LMS J. Comput. Math.*, vol. 15, pp. 38–43, Apr. 2012.

[32] E. Psomopoulos, "Commutativity theorems for rings and groups with constraints on commutators," *Int. J. Math. Math. Sci.*, vol. 7, no. 3, pp. 513–517, 1984.

[33] I. F. Rúa, E. F. Combarro, and J. Ranilla, "Classification of semifields of order 64," *J. Algebra*, vol. 322, no. 11, pp. 941–961, 2009.

[34] I. F. Rúa, E. F. Combarro, and J. Ranilla, "Determination of division algebras with 243 elements," *Finite Fields Appl.*, vol. 18, pp. 1148–1155, Nov. 2012.

[35] S. Satyajit, K. Srinivasan, B. K. Behera, and P. K. Panigrahi, "Nondestructive discrimination of a new family of highly entangled states in IBM quantum computer," *Quantum Inf. Process.*, vol. 17, no. 9, p. 212, 2018.

[36] V. V. Shende and I. L. Markov, "On the CNOT-cost of TOFFOLI gates," *Quantum Inf. Comput.*, vol. 9, no. 5, pp. 461–486, 2009.

[37] R. D. Schafer, "An introduction to nonassociative algebras," in *Pure and Applied Mathematics*, vol. 22. New York, NY, USA, 2011.

[38] P. W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in *Proc. FOCS*, 1994, pp. 124–134.

[39] P. W. Shor, "Why haven't more quantum algorithms been found?" *J. ACM*, vol. 50, no. 1, pp. 87–90, 2003.

[40] A. C.-C. Yao, "Probabilistic computations: Toward a unified measure of complexity," in *Proc. 18th IEEE Symp. Found. Comput. Sci. (FOCS)*, Oct. 1977, pp. 222–227.

ELÍAS F. COMBARRO received the B.S. degree in mathematics, the M.S. degree in computer science, and the Ph.D. degree in mathematics from the University of Oviedo, Oviedo, Spain, in 1997, 2001, and 2002, respectively, where he is currently an Associate Professor.

He has authored more than 30 research papers in topics such as computability theory, the theory of fuzzy measures, and the computational classification of semifields and text categorization. His current research interest includes quantum computing.

JOSÉ RANILLA received the B.S. and M.S. degrees in computer science from the Polytechnic University of Valencia, Valencia, Spain, in 1987, and 1993, respectively, and the Ph.D. degree in computer science from the University of Oviedo, Oviedo, Spain, in 1998, where he is currently a Professor.

He has authored more than 80 research papers in topics such as high-performance and parallel computing, and the computational classification of semifields and text categorization. His current research interest includes quantum computing.

IGNACIO F. RÚA received the B.S., M.S., and Ph.D. degrees in mathematics from the University of Oviedo, Oviedo, Spain, in 1999, 2001, and 2004, respectively, where he is currently an Associate Professor.

From 2004 to 2007, he was a Research Fellow of the Spanish Juan de la Cierva Program with the Universidad de Cantabria. He has coauthored 30 research papers on non-associative finite rings and their applications in coding theory and cryptography. His current research interests include computer algebra and quantum computing.