Quantum Information and Computation, Vol. 23, No. 15&16 (2023) 1275–1290
© Rinton Press

EFFICIENT QUANTUM ALGORITHMS TO FIND SUBSTRUCTURES ON FINITE ALGEBRAS

J.M. HERNÁNDEZ CÁCERES
Department of Mathematics, University of Oviedo, jmhernandez@uniovi.es
Oviedo 33007, Spain

I.F. RÚA
Department of Mathematics, University of Oviedo, rua@uniovi.es
Oviedo 33007, Spain

ELíAS F. COMBARRO
Department of Computer Science, University of Oviedo, efernandezca@uniovi.es
Oviedo 33005, Spain

Received April 27, 2023
Revised November 23, 2023

When classifying a collection of finite algebras (for instance, in the computational classification of finite semifields), an important task is the determination of substructures such as the right, middle, and left nuclei, the nucleus, and the center. Finding these structures may become computationally expensive when there is no additional information about the algebra properties. In this paper, we introduce quantum algorithms than solve this task efficiently, by formulating it as an instance of the Hidden Subgroup Problem (HSP) over Abelian groups. We give detailed constructions of the quantum circuits involved in the process and prove that the overall (quantum) complexity of our algorithm is polynomial in the dimension of the algebra, while a similar approach with classical computers would require an exponential number of queries to the HSP function.

Keywords: Hidden Subgroup Problem, Finite Semifield, Nuclei, Center, Quantum Algorithms.

1 Introduction

Quantum computing is a computational model based on exploiting quantum phenomena such as superposition, interference and entanglement. It is well-known that some quantum algorithms outperform their classical counterparts, at least with current knowledge or under oracular assumptions. For instance, based on the quantum Fourier transform (perhaps one of the most important unitary transformation in quantum computing), factoring integers can be solved in polynomial time by Shor's algorithm [1], while no classical algorithm solving the same problem with such a complexity is known. This algorithm can be regarded under the framework of the hidden subgroup problem, as the computation of the order of an element in a cyclic group is equivalent to finding one of its subgroups.

Hidden Subgroup Problem: Let G be a finite group, and let $H \subseteq G$ be one of its subgroups. Let S be a set, and let $g : G \to S$ be a function that distinguishes cosets of H, i.e., for all $g_1, g_2 \in G, f(g_1) = f(g_2) \Leftrightarrow g_1H = g_2H$. The hidden subgroup problem (HSP) is to determine a generating set for the subgroup H, given access to a black box that evaluates f on arbitrary elements.

For specific groups, Abelian for instance, efficient quantum algorithms solving the HSP are known. In this paper, we want to use this fact to efficiently compute substructures of finite algebras, and in particular of finite semifields. So, let K be a finite field, and let A be a non-associative and non-commutative finite-dimensional K-algebra with a fixed basis $\beta = \{e_1, \ldots, e_n\}$. There exists a unique set of constants $\{M_{ijk}\}_{i,j,k=1}^n \subseteq K$ such that $e_i \cdot e_j = \sum_{k=1}^n M_{ijk} e_k$, for all $i, j \in \{1, \ldots, n\}$. That set is known as the multiplication table of the algebra. Consider the additive group $(A, +)$, i.e., the elements of the algebra with the addition operation. Given two elements $a, b \in A$, we define the commutator $[a, b] = ab - ba$, which measures the non-commutativity of A. For instance, A is a commutative algebra if and only if $[e_i, e_j] = 0$ for all $i, j = 1, \ldots, n$. (It is worth to mention that quantum procedures for testing the commutativity of a finite dimensional algebra have been proposed [2]). We define the associator as a multilinear map $[\cdot, \cdot, \cdot] : A \times A \times A \to A$ given by $[x, y, z] = (xy)z - x(yz)$. It measures the non-associativity of A, so if A is associative then the associator is identically zero. Consider the following sets, known as the right, middle, and left nuclei, the nucleus and the center:

$$\begin{aligned} N_r(A) &= \{a \in A : [x, y, a] = 0 \text{ , for all } x, y \in A\}, \\ N_m(A) &= \{a \in A : [x, a, y] = 0 \text{ , for all } x, y \in A\}, \\ N_l(A) &= \{a \in A : [a, x, y] = 0 \text{ , for all } x, y \in A\}, \\ N(A) &= N_r(A) \cap N_m(A) \cap N_l(A), \\ Z(A) &= N(A) \cap \{a \in A : [a, x] = 0 \text{ , for all } x \in A\}. \end{aligned} \tag{1}$$

These sets which can be written in terms of the K-basis β and the multiplication table, provide information about the algebra. For instance, when A is a finite semifield, i.e., a finite division ring, these sets are related to properties of the corresponding coordinates projective planes [3]. Finding those sets can be stated in terms of the HSP, and it is clearly important in the context of the classification of finite semifields, see for instance [4], and [5]. Our problem is, then, stated as follows:

Given: Multiplication table of a finite dimensional K-algebra A. (K finite field)
Problem: Find $N_r(A), N_m(A), N_l(A), N(A)$ and $Z(A)$.

In order to solve it with quantum techniques, we will transform each problem of finding $N_r(A), N_m(A), N_l(A), N(A)$ and $Z(A)$ into a instance of the HSP, which in general can be stated as follows:

Given: The ability to evaluate a hiding function f for a subgroup H of a group G (i.e., a function f that is constant on a subgroup H of $G = (A, +)$, and is distinct on different cosets of H) on arbitrary elements of A.

Problem: To find $s_1, s_2, \ldots, s_l$, a generating set for H.

In this work, we explicitly and efficiently construct quantum circuits that, from the multiplication table of the n-dimensional algebra over a finite field $\mathbb{F}_p$, implement functions hiding function f that can be used to determine these sets using only a polynomial number of quantum gates, in fact of order $O(n^5r^3)$, with $O(nr)$ queries to the oracle to find those sets, where $r = \lceil \log_2(p) \rceil$ (i.e., with an asymptotically linear number of evaluations of the function f). This is achieved by suitable choices of functions f in the previous problem and by then using the quantum solution for the HSP over finite Abelian groups (see [6], for instance). Also, we prove that this can not be achieved classically with a polynomial number of accesses to the function f if given only access to a black box oracle to evaluate f, without additional information on the algebra.

The structure of the paper is as follows. In Section 2, we collect the basic results needed for our work, including a short background on Quantum Computing. In Section 3, we model the problem of finding substructures in a finite-dimensional algebra as an instance of the HSP, and we show that it, in some cases, it can not be classically solved with a polynomial number of function accesses to the hiding function f. In Section 4, we construct an efficient quantum oracle for the function f, and we build an efficient circuit for the solution of the corresponding HSP. In Section 5, we illustrate the use of the algorithm by applying it to two simple examples. Finally, in Section 6, we give some conclusions of our study and propose some ideas for future work.

2 Quantum preliminaries

The quantum circuit model is one of the most popular models for quantum computing. In it, qubits store data, operations are performed with quantum gates, and results are obtained via measurements. All of these are ruled by the laws of quantum mechanics and explicitly use quantum properties such as superposition, interference and entanglement, making this model quite different from classical ones.

Let us collect some basic notions on qubits and quantum gates that would be useful for the next section and the rest of our discussion. Details can be found for instance in [7, 8]. Let us begin with some notation related to the states of a quantum computation.

2.1 Quantum states.

As it is customary in quantum computing, we will use Dirac notation, where row and column vectors are represented using bras and kets. A *ket* is a term of the form $|v\rangle$. Mathematically it denotes a vector $v \in \mathbb{C}^n$. The *bra* of a vector $v \in \mathbb{C}^n$ denoted by $\langle v|$, is defined as $\langle v| = (|v\rangle)^\dagger$, where $\dagger$ denotes the conjugate transpose.

Dirac notation is useful to represent the state of quantum systems. The simplest case is that of a qubit, or a quantum state with two possible measurement results. In contrast to normal bits, which can just be in state 0 or in state 1, a quantum-bit or qubit for short, can be in state $|0\rangle = \begin{bmatrix} 1 & 0 \end{bmatrix}^t$ or $|1\rangle = \begin{bmatrix} 0 & 1 \end{bmatrix}^t$, (where t denotes the transpose) or it could be in a superposition $\alpha|0\rangle + \beta|1\rangle$, where $\alpha, \beta \in \mathbb{C}$ such that $|\alpha|^2 + |\beta|^2 = 1$, are called the amplitudes of the state. All of these possible values for the state of a single qubit are normalized states of vectors in $\mathbb{C}^2$, for which $B = \{|0\rangle, |1\rangle\}$ is an orthonormal basis, known as computational.

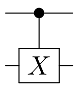
The states of quantum systems composed of several qubits can be represented by vectors in tensor products of $\mathbb{C}^2$. The computational basis for $\mathbb{C}^{2^n} \cong (\mathbb{C}^2)^{\otimes n}$ is the tensor product of the basis B, namely $\{|0\rangle \otimes |0\rangle \otimes \cdots \otimes |0\rangle, |0\rangle \otimes |0\rangle \otimes \cdots \otimes |1\rangle, \ldots, |1\rangle \otimes |1\rangle \cdots \otimes |1\rangle\} = \{|0\rangle, |1\rangle, \ldots, |2^n - 1\rangle\}$. So, a generic state of a multi-qubit system is $|\psi\rangle = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle$ where $\sum_{i=0}^{2^n-1} |\alpha_i|^2 = 1$ and $\alpha_i \in \mathbb{C}$ for $0 \le i \le 2^n - 1$.

2.2 Quantum gates.

In order to perform computations, we need to manipulate the state of qubits. In the quantum circuit model, operations correspond to matrices U that are unitary (i.e., they satisfy $U^\dagger U = UU^\dagger = I_n$, where I_n is the identity matrix of size $2^n \times 2^n$), and they are called quantum gates. Some important one-qubit quantum gates are

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \tag{2}$$

where H is the Hadamard gate (that can be use to put a basis state in superposition), and the X gate (which is the quantum version of the classical NOT gate). An example of a two-qubit gate (arguably the most important one is the CNOT, or controlled-NOT, given by $|0\rangle\langle 0| \otimes I_2 + |1\rangle\langle 1| \otimes NOT$, which flips the second qubit when the first one is equal to $|1\rangle$. The CNOT gate in a quantum circuit is represented as:



(3)

In a quantum circuit, the horizontal lines are called wires and they represent the qubits that we are working with, and the circuit is read from left to right. Two one-qubit gates U_1, U_2 acting on each qubit $|\psi_1\rangle, |\psi_2\rangle$ independently yield a two-qubit gate whose action is given by $(U_1 \otimes U_2)(|\psi_1\rangle \otimes |\psi_2\rangle) = U_1 |\psi_1\rangle \otimes U_2 |\psi_2\rangle$. In general, we can construct a n-qubit gate U as $U = U_1 \otimes U_2$ where U_1 is a n_1-qubit gate, U_2 is a n_2-qubit gate, and $n = n_1 + n_2$. Observe that the $n-$tensor product of Hadamard gates acts as

$$H^{\otimes n} |x\rangle = \frac{1}{\sqrt{2^n}} \sum_{j=0}^{n} (-1)^{x \cdot j} |j\rangle. \tag{4}$$

where $x \in \{0,1\}^n$, and $x \cdot j = x_1 j_1 \oplus x_2 j_2 \oplus \cdots \oplus x_n j_n$ (i.e., addition mod 2), i.e., it provides superposition of qubits and so it is convenient to use it at the beginnning of a quantum circuit to achieve a state of uniform superposition of all the basis states.

It is worth mentioning, that there are unitary matrices that cannot be written as the tensor product of unitary matrices. This is the case, for example, of the CNOT gate and of the three-qubit Toffoli gate, that acts on a basis state $|x\rangle |y\rangle |z\rangle$ taking it to $|x\rangle |y\rangle |z \oplus x \wedge y\rangle$, where $\oplus$ is addition modulo 2 and $\wedge$ is the logical AND. Notice that we can see the Toffoli gate as doubly controlled NOT gate.

2.3 Quantum Fourier Transform over Abelian Groups.

One of the most important unitary transformation in quantum computing is the *Quantum Fourier Transform (QFT)*. Here, we only consider in the special case of the Abelian group

$A = (\mathbb{Z}/p\mathbb{Z})^n$. Let $r = \lceil \log_2(p) \rceil$, and consider that any element $g \in A$ is associated to a computational basis element $|g\rangle$ of $(\mathbb{C}^2)^{\otimes rn}$. Then, the QFT over G is the transformation with the following action on the basis states,

$$F_G |g\rangle = \frac{1}{\sqrt{p^n}} \sum_{h \in A} \omega_p^{gh} |h\rangle \tag{5}$$

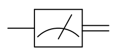
where, $\omega_p = \exp\left(\frac{2\pi}{p}\right)$ is a primitive $p-$th root of unity, and gh denotes the inner product $\sum_{i=1}^n g_i h_i \bmod p$.

2.4 Oracles.

Many quantum algorithms are based around the analysis of some function f. An oracle (also called black box) is a special kind of unitary transformation that is defined by its action on the computational basis and that is given as a gate that can be used in a quantum circuit but whose inner workings cannot be inspected. One of the main forms that oracles take is that of Boolean oracles, for a Boolean function $f : \{0,1\}^n \to \{0,1\}^m$. Such a unitary operator, denoted U_f, is defined by $U_f |a\rangle |h\rangle = |a\rangle |h \oplus f(a)\rangle$, and it acts on quantum states in $(\mathbb{C}^2)^{\otimes(n+m)}$ ($\oplus$ denotes the bitwise exclusive OR, $a \in \{0,1\}^n$ and $h \in \{0,1\}^m$). Observe that, when the second $m-$qubit register is $|0\rangle$, application of U_f yields an evaluation of the content of the first $n-$qubit register.

2.5 Measurements.

The measurement operator allows us to obtain classical values from quantum states. It is denoted by the gauge symbol:



(6)

Measurement of a quantum state in superposition yields its collapse into one of the computational basis states, with a given probability. For example, if we measure $\alpha |0\rangle + \beta |1\rangle$ with $\alpha, \beta \in \mathbb{C}$ and $|\alpha|^2 + |\beta|^2 = 1$, we obtain the quantum state $|0\rangle$ with probability $|\alpha|^2$, and the quantum state $|1\rangle$ with probability $|\beta|^2$. After such a measurement, the qubit collapses into the measured state. In general, if we have a state

$$\sum_{x \in \{0,1\}^n, y \in \{0,1\}^m} a_{xy} |x\rangle |y\rangle , \tag{7}$$

and we measure the first $n-$qubit register, we will obtain $|x_0\rangle$ with probability

$$\sum_{y \in \{0,1\}^m} |a_{x_0 y}|^2, \tag{8}$$

and the quantum state then collapses to

$$\frac{\sum_{y \in \{0,1\}^m} a_{x_0 y} |x_0\rangle |y\rangle}{\sqrt{\sum_{y \in \{0,1\}^m} |a_{x_0 y}|^2}}. \tag{9}$$

3 The classical approach

In this section, we model the problem of finding substructures in a finite-dimensional algebra as an instance of the HSP, and we show that, in general, it can not be classically solved in a number of function accesses to the hiding function f that is polynomial in the dimension of the algebra. In particular, we explicitly give the functions that hide the right, middle, and left nuclei, nucleus and center of a finite dimensional K-algebra A, in terms of its multiplication table.

3.1 Hiding functions

Namely, for N_r consider the following function:

$$\begin{array}{rccc} f_{N_r}: & A & \to & A^{n^2} \\ & a & \mapsto & f_{N_r}(a) = ([e_1,e_1,a],[e_1,e_2,a],\ldots,[e_n,e_n,a]) \end{array} \tag{10}$$

Note that $f_{N_r}(a_1) = f_{N_r}(a_2)$ if and only if $a_1 - a_2 \in N_r(A)$. Indeed,

$$\begin{aligned} f_{N_r}(a_1) = f_{N_r}(a_2) &\Leftrightarrow ([e_1,e_1,a],\ldots,[e_n,e_n,a]) = ([e_1,e_1,a],\ldots,[e_n,e_n,a]) \\ &\Leftrightarrow ([e_1,e_1,a_1]-[e_1,e_1,a_2],\ldots,[e_n,e_n,a_1]-[e_n,e_n,a_2]) = (0,\ldots,0) \\ &\Leftrightarrow [e_i,e_j,a_1]-[e_i,e_j,a_2] = 0 \text{ for all } i,j=1,\ldots,n. \\ &\Leftrightarrow (e_ie_j)\,a_1 - e_i\,(e_ja_1) - (e_ie_j)\,a_2 + e_i\,(e_ja_2) = 0 \text{ for all } i,j=1,\ldots,n. \\ &\Leftrightarrow (e_ie_j)\,(a_1-a_2) - e_i\,(e_j\,(a_1-a_2)) = 0 \text{ for all } i,j=1,\ldots,n. \\ &\Leftrightarrow (a_1-a_2) \in N_r(A). \end{aligned} \tag{11}$$

Hence, we can say that f hides the subgroup $N_r(A)$. For $N_m(A), N_l(A)$ we consider the functions

$$\begin{array}{rccc} f_{N_m}: & A & \to & A^{n^2} \\ & a & \mapsto & f_{N_m}(a) = ([e_1,a,e_1],[e_1,a,e_2],\ldots,[e_n,a,e_n]) \end{array} \tag{12}$$

and

$$\begin{array}{rccc} f_{N_l}: & A & \to & A^{n^2} \\ & a & \mapsto & f_{N_l}(a) = ([a,e_1,e_1],[a,e_1,e_2],\ldots,[a,e_n,e_n]) \end{array} \tag{13}$$

Analogously, as in the case of f_{N_r}, it can be seen that $f_{N_m}(a_1) = f_{N_m}(a_2)$ if and only if $a_1 - a_2 \in N_m(A)$, and that $f_{N_l}(a_1) = f_{N_l}(a_2)$ if and only if $a_1 - a_2 \in N_l(A)$. Hence, we can say that f_{N_m} and f_{N_l} hide the subgroups $N_m(A)$ and $N_l(A)$, respectively. For $N(A)$ consider the function

$$\begin{array}{rccc} f_N: & A & \to & A^{3n^2} \\ & a & \mapsto & f_N(a) = (f_{N_r}(a), f_{N_m}(a), f_{N_l}(a)) \end{array} \tag{14}$$

So, $f_N(a_1) = f_N(a_2)$ if and only if $a_1 - a_2 \in N_r(A) \cap N_m(A) \cap N_l(A) = N(A)$. And, for $Z(A)$ consider the following function

$$\begin{array}{rccc} f_Z: & A & \to & A^{3n^2+n} \\ & a & \mapsto & f_Z(a) = (f_N(a),[a,e_1],\ldots,[a,e_n]) \end{array} \tag{15}$$

Hence, $f_Z(a_1) = f_Z(a_2)$ if and only if $a_1 - a_2 \in N(A) \cap \{a \in A : [a,e_i] = 0 \text{ for } i = 1,\ldots,n\} = Z(A)$.

As we can see, all hiding functions are given in terms of commutators and associators of the basis elements of the algebra and the argument of the hiding function. So, in order to show that the hiding functions can be written in terms of the multiplication table of the algebra, we only need to observe that if $a = \sum_{m=1}^{n} \alpha_m e_m$, then we have the conmutator $[a, e_i] = \sum\limits_{k=1}^{n} \left(\sum\limits_{m=1}^{d} \alpha_m \left(M_{mik} - M_{imk} \right) \right) e_k$ and the associator $[e_i, e_j, a] = \sum\limits_{r=1}^{d} \left(\sum\limits_{m=1}^{d} \alpha_m \left(\sum_{k=1}^{d} \left(M_{ijk} M_{kmr} - M_{jmk} M_{ikr} \right) \right) \right) e_r$ Analogously, for the associator $[e_i, a, e_j]$ and $[a, e_i, e_j]$.

3.2 Classical solution

Next, we consider the classical (i.e. non-quantum) solution to the HSP. The idea is to show that as long as there exist different subgroups hidden by the same function, extra evaluations of the hiding function are needed in order to distinguish them. The following is an auxiliary technical result.

Lemma 1: Let G be a finite group having N subgroups with trivial pairwise intersection. Let $g_1, \ldots, g_t \in G$ be such that $m = N - \binom{t}{2} \geq 1$. Then, there exist m subgroups $H_1, \ldots, H_m$ out of the N, given with hiding functions $f_1, \ldots, f_m : G \to \mathbb{N}$, such that $f_i(g_k) = f_j(g_k)$ for all $1 \leq i, j \leq m$, and $1 \leq k \leq t$.

Proof: The proof follows by induction over t. For $t = 1$, define $f_i(g_1) = 1$ for every one of the $i = 1, \ldots, N$ subgroups, and extend to G in the following way: $f_i(g) = j$ if $gH = e_j H$, where $g_1 H_i, e_2 H_i, \ldots, e_{t_i} H_i$ are the different classes of G mod H_i.

Assume for $t > 1$ that there exist $m = N - \binom{t-1}{2}$ subgroups $H_1, \ldots, H_m$ with hiding functions $f_1, \ldots, f_m$ such that $f_i(g_k) = f_i(g_k)$ for all $1 \leq i, j \leq m$ $1 \leq k \leq t-1$. Take $g_t \in G$. If there exist a $k < t$ such that $g_t = g_k$ the result follows directly. Otherwise, $g_t \notin \{g_1, \ldots, g_{t-1}\}$, and consider $h_k = g_t^{-1} g_k \neq 1$ for $k = 1, \ldots, t-1$. A fixed h_k belongs, at most, to 1 of the $H_1, \ldots, H_m$ subgroups (because $h_k \neq 1$). Take those H_l out of the N given such that $h_k \notin H_l$ for $k = 1, \ldots, m$. We can redefine $f_l(g_t h_l) = \max\{f_l(g_k)\} + 1$, for all $h_l \in H_l$. This still can be seen to hide H_l, because $g_t^{-1} g_k \notin H_l$ for all $k = 1, \ldots, m$. At most, there are $t-1$ subgroups that are not taken in this step, so we are left with

$$m - (t-1) = N - \binom{t-1}{2} - (t-1) = N - \binom{t}{2} \tag{16}$$

thus the result follows. $\square$.

Theorem 2: Under the conditions of the previous lemma, t evaluations of a hiding function f are not enough to solve the corresponding instance of the HSP in a classical computer. This holds, in particular, if $t \leq \lfloor \sqrt{N} \rfloor$ and $N \geq 2$.

Proof:

$$\binom{t}{2} < \frac{t^2}{2} \leq \frac{N}{2} \tag{17}$$

Then

$$N - \binom{t}{2} = \frac{N}{2} + \frac{N}{2} - \binom{t}{2} \geq \frac{N}{2} + 1 \geq 2 \tag{18}$$

Assume the existence of a HSP solver using at most t evaluations of a hiding function. Evaluations in the t elements $g_1, \ldots, g_t$ of the previous theorem provide the same information for the m subgroups $H_1, \ldots, H_m$. Consequently, they can not be distinguished by the HSP solver. $\square$.

Corollary 3: Computing the right, middle, and left nuclei, center and center of a finite dimensional algebra A over a finite field K of q elements, with a HSP solver on a classical computer requires at least $\Omega(\sqrt{q^n})$ evaluations of the corresponding hiding function.

Proof: In our case, if we consider the additive group of the algebra $G = (A, +) \cong K^n$, the number of $1-$dimensional subspaces is $1 + q + \cdots + q^{n-1}$, all of them with trivial pairwise intersection. Following the corollary, it is necessary at least $\lfloor \sqrt{1 + q + \cdots + q^{n-1}} \rfloor = \Omega(\sqrt{q^n})$ evaluations for a HSP solver to compute each of the mentioned substructures. $\square$.

As an aside note, notice that this kind of situation can arise in practice, even when dealing with semifields. For instance, observe that it is always possible to have a finite dimensional algebra A for which $N_l = N_m = N_r = N = Z$ is of dimension 1 over K (for instance, a Generalized Twisted Field for particular choice of its defining parameters [9]).

4 The quantum approach

In this section, we solve the problem of finding substructures in a finite-dimensional algebra by quantum procedures. In particular, we construct an efficient quantum oracle for the hiding functions of the substructures, and we build an efficient circuit for the solution of the corresponding HSP. Let us first give an overall picture of the whole procedure.

Algorithm 1: [Solution to the HSP for substructures of a finite algebra]
Input: Multiplication table of a finite dimensional K-algebra A with respect to a basis $\{e_1, \ldots, e_m\}$ (K finite field of p elements)
Construction of quantum oracle: From the multiplication table, construct a quantum oracle for the hiding function f of a specific subgroup H.
Quantum procedure: Construct of a quantum circuit that, using the quantum oracle, returns tuples $(\alpha_1, \ldots, \alpha_m)$ of elements in K, which provide cooordinates of elements in the orthogonal complement of H, i.e., $(\alpha_1, \ldots, \alpha_m) \cdot (\beta_1, \ldots, \beta_m) = 0$, for all $\sum_{i=1}^{m} \beta_i e_i \in H$.
Classical post-processing: Compute generators of H from Gaussian elimination on the tuples given by the quantum procedure.

4.1 Oracle of the hiding function

Our first task is to show that an efficient quantum oracle (in terms of number of quantum gates) can be constructed from the multiplication table of the algebra, for each of the hiding functions introduced in the previous section.

We consider the construction for $N_r(A)$, as $N_m(A), N_l(A), N(A), Z(A)$, follow the same lines. From the previous section, $f_{N_r}(a)$ can be written in terms of the structure constants, since if $a = \sum_{m=1}^{n} \alpha_m e_m$, then for all $i, j = 1, \ldots, n$,

$$\begin{aligned}[e_i, e_j, a] &= \sum_{s=1}^{n} \left(\sum_{m=1}^{n} \left(\sum_{k=1}^{n} (M_{ijk} M_{kms} - M_{jmk} M_{iks})\, \alpha_m \right) \right) e_s \\ &= \sum_{s=1}^{n} \left(\sum_{m=1}^{n} \alpha_m \lambda_{ijms} \right) e_s\end{aligned} \tag{19}$$

where $\lambda_{ijms} = \sum\limits_{k=1}^{n} (M_{ijk} M_{kms} - M_{jmk} M_{iks})$ are n^4 constants in K. If we consider the coordinates of such an expression, we can go further and expand it to get a coordinate version of f_{N_r} namely

$$\left(\sum_{m=1}^{n} \alpha_m \lambda_{11m1}, \ldots, \sum_{m=1}^{n} \alpha_m \lambda_{ijms}, \ldots, \sum_{m=1}^{n} \alpha_m \lambda_{nnmn} \right) \tag{20}$$

all λ_{ijms}, for $i, j, m, s = 1, \ldots, n$, together with their binary representations $\{\lambda_{ijms}^{t}\}$ for $t = 1, \ldots, r = \lceil \log_2(p) \rceil$. Using the controlled-multiplier modulo p of [10], each $\lambda_{ijms}\alpha_m$ can be implemented by repeated modular additions (modulo p), requiring $3r+1$ qubits, and $O(r^2)$ gates (among NOT, CNOT and Toffoli gates). This gives an overall number of $O(nr)$ qubits and $O(nr^2)$ gates, for each of the products involved in the modular addition $\sum_{m=1}^{n} \lambda_{ijms}\alpha_m$. Again by [10], such a modular addition can be carried out with $O((n-1)r)$ gates. Since the final result would be storaged only on r qubits, the $(n-1)r$ remaining ones can be reused. Therefore, $\sum_{m=1}^{n} \alpha_m \lambda_{ijms}$ requires $O(nr^2)$ gates and $O(nr)$ qubits. Since there are n^3 sums of that form, the U_{N_R} oracle can be efficiently constructed with $n^3 r + nr$ qubits and $4r+1$ ancillary qubits. In short, the oracle acts as $U_{N_r} |a\rangle |0\rangle = |a\rangle |f_{N_r}(a)\rangle$ for $a \in A, h \in A^{n^2}$.

In summary, in Table 1, we give the number of qubits corresponding to inputs, outputs, ancillary, and the order of the number of gates required to build each oracle $U_{N_r}, U_{N_m}, U_{N_l}, U_N, U_Z$.

Table 1. Cost in terms of number of qubits and gates of each oracle.

Oracle		$U_{N_r}, U_{N_m}, U_{N_l}$	U_N	U_Z
Number of qubits	Input	nr		
	Output	$n^3 r$	$3n^3 r$	$3n^3 r + n^2 r$
	Ancillary	$4r+1$		
Number of gates		$O(n^4 r^2)$		

4.2 Quantum Algorithm To Find Substructures

Next, we present a quantum efficient algortihm to find the substructures of a finite-dimensional algebra over a finite field, based on the above constructed quantum oracle and the ideas of [6, Section 3]. It uses an efficient computation of the quantum Fourier transform on $G = (A, +) \cong (\mathbb{Z}/p\mathbb{Z})^n$, and it outputs elements of the orthogonal subgroup H to be found:

$$H^{\perp} = \{\alpha \in G : \alpha_1\beta_1 + \cdots + \alpha_n\beta_n 0 \equiv \text{mod } p, \text{ for all } \beta \in H\} \tag{21}$$

As before, we consider the construction for $N_r(A)$, as that of $N_m(A), N_l(A), N(A)$, $Z(A)$, follow the same lines.

Algorithm 2:[Quantum procedure for finding the orthogonal of the Right Nucleus.]

Input: A black box which performs the operation $U_{N_r} \left| a \right\rangle \left| 0 \right\rangle = \left| a \right\rangle \left| f_{N_r}(a) \right\rangle$ for $a \in A$.

Quantum Procedure:

1. Initial state: $\left| 0 \right\rangle^{\otimes r_1} \left| 0 \right\rangle^{\otimes r_2} \left| 0 \right\rangle^{\otimes r_3}$, with $r_1 = nr$ input qubits, $r_2 = n^3 r$ output qubits plus $r_3 = 4r + 1$ ancillary qubits ($r = \lceil \log_2(p) \rceil$).

2. Create superposition and remove elements which are $\geq p$

3. Apply the black box U_{N_r}

4. Apply the Quantum Fourier Transform on the first register.

5. Measure the first register.

Output: The binary expansion of the coordinates of an element in $N_r(A)^{\perp}$.

As in many quantum algorithms, superposition is achieved by applying the Hadamard transformation. However, we must notice that removal of elements greater or equal than p is needed, as we are only interested in values mod p. After that, as it is standard in quantum solutions to the HSP, an application of the quantum oracle U_{N_r} is followed by a QFT and a measurement.

4.2.1 Superposition and removal of elements greater or equal than p.

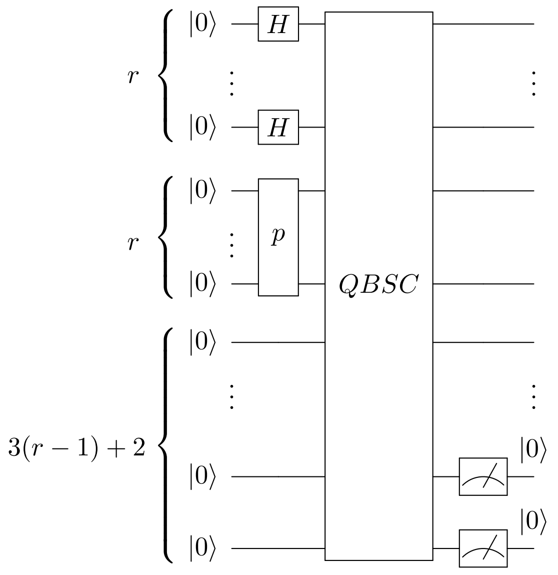
Let us explain with a litte more detail the second step of the quantum procedure. Each element in the algebra is represented by the binary expansion of its coordinates. So, we need nr qubits to deal with all elements in the algebra in superposition. Let us divide then in n registers of r qubits. Each of then encoding a single coordinate, which is an integer mod p. Therefore, since we are only interested in a superposition of p constants ($\frac{1}{\sqrt{p}} \sum_{x=0}^{p-1} \left| x \right\rangle$), after a standard superposition of the r qubits with Hadamard gates ($\frac{1}{\sqrt{2^r}} \sum_{x=0}^{2^r-1} \left| x \right\rangle$), we need to remove those sumands which are greater or equal than p.

This is accomplished by the use of an extra r qubit register, storing the binary representation of the integer p (by an application of at most r X gates). We represented it on the circuit of Figure 1 as

$$-\boxed{p}- \qquad (22)$$

Now, for each of the n pairs of $2r$ qubit registers, we use the quantum bit string comparator (QBSC) from [11] to remove the undesirable summands (see Figure 1). It has a total cost of $O(r)$ CNOT and single qubit gates, and $3r-1$ ancillary qubits. The last two of them, Q_1, Q_2, provide after measurement a comparison with p. Namely, the integer is smaller than p if and only if $Q_1 = 1$ and $Q_2 = 0$. So, such a measurement yields the collapse of the first register to the desired superposition. An undesired measurement forces a repetition of the process.

The probability of failure of one single comparison is $\frac{2^r - p}{2^r} < \frac{1}{2}$. Therefore, if the process is repeated t times, the probability of failure is at most $\frac{1}{2^t}$. Since this technique is to be applied in parallel to each of the n different $r-$qubit registers, the overall probability of failure is at most $\frac{n}{2^t}$. Choosing $t = \log_2\left(\frac{n}{\varepsilon}\right) + 1 = O(\log(n))$, yields a bounded probability error of the whole comparison procedure $0 < \varepsilon < 1$, that can be made arbitrarily small.



(23)

Fig. 1. Quantum circuit of the step 2 of the quantum procedure.

Fix an ϵ with $0 < \epsilon < 1$. Let $t_1 = \lfloor \log_2 \left(\frac{n}{\epsilon}\right) \rfloor + 1$. The number of ancillary qubits required in a single use of a QBSC is $3(r-1)+2$ and the number of qubits measured are two. However, we can uncompute the $3(r-1)$ qubits that were not measured and reset the measured ones. We apply this process t_1 times sequentially for each r qubits from the nr qubits of the first register, giving an overall of $3(r-1)+2$ ancillary qubits. Now, adding the r qubits from the state of p to the ancillary we have $3(r-1)+2+r = 4r-1$.

4.2.2 Steps 3 to 5 in the algorithm.

Assuming the previous step is successful, we achieve the quantum state

$$\frac{1}{\sqrt{p^n}} \sum_{a \in A} |a\rangle |0\rangle^{\otimes r_2} \tag{24}$$

(ancillary qubits are omited) and application of the oracle U_{N_r} yields

$$\frac{1}{\sqrt{p^n}} \sum_{a \in A} |a\rangle |f_{N_r}(a)\rangle \tag{25}$$

Application of the QFT on the first register gives

$$\sum_{b \in A} |b\rangle \left(\frac{1}{p^n} \sum_{a \in A} \omega_p^{ab} |f_{N_r}(a)\rangle \right) \tag{26}$$

where $\omega_p = \exp\left(\frac{2\pi i}{p}\right)$. A measurement $|b\rangle$ on the first register occurs with probability

$$\left\| \frac{1}{p^n} \sum_{a\in A} \omega_p^{ab} \left| f_{N_r}(a) \right\rangle \right\|^2 \tag{27}$$

If $\{s_1, s_2, \ldots, s_l\}$ is a $\mathbb{Z}/p\mathbb{Z}-$basis of H, then for all $z \in \text{Range}(f_{N_r})$, there exists $a_z \in A$ such that $z = f_{N_r}(a_z + \sum_{i=1}^{l} \lambda_i s_i)$, for all $0 \leq \lambda_1, \ldots, \lambda_l \leq p-1$. So, the probability becomes

$$\begin{aligned} &\left\| \frac{1}{p^n} \sum_{z\in \text{Range}(f_{N_r})} \left(\sum_{\lambda_1,\ldots,\lambda_l=0}^{p-1} \omega_p^{b(a_z + \sum_{i=1}^{l} \lambda_i s_i)} \right) |z\rangle \right\|^2 \\ =&\left\| \frac{1}{p^n} \sum_{z\in \text{Range}(f_{N_r})} w_p^{ba_z} \prod_{i=1}^{l} \left(\sum_{\lambda=0}^{p-1} \omega_p^{b\lambda s_i} \right) |z\rangle \right\|^2 \end{aligned} \tag{28}$$

Now, when $bs_i \neq 0 \bmod p$ for some $i = 1, \ldots, l$, then $\sum_{\lambda=0}^{p-1} \omega_p^{\lambda b s_i} = 0$ (because $w_p^{bs_i}$ is a $p-$th primitive root of unity, and so it is a root of the $p-$th cyclotomic polynomial $\sum_{\lambda=0}^{p-1} x^\lambda$), and so the corresponding summand vanishes. Otherwise, $b \in H^\perp$, and the corresponding probability is

$$\frac{1}{p^{2n}} \sum_{z\in \text{Range}(f_{N_r})} \left| w_p^{ba_z} \prod_{i=1}^{l} \left(\sum_{\lambda=0}^{p-1} \omega_p^0 \right) \right|^2 = \frac{1}{p^{2n}} p^{n-l} (1\cdot p^l)^2 = p^{l-n} \tag{29}$$

Thus, we obtain an element uniformly distributed of $N_r(A)^\perp$.

4.3 Classical post processing

In order to determine the number of times that the quantum procedure should be run in order to find a generator set of the subgroup $N_r(A)^\perp$, we shall use

Theorem 4:[6, Theorem D.1] Let G be a finite group. For an integer $t \geq 0$, the probability that $k = t + \log_2 |G|$ elements chosen uniformly at random from G will generate G is at least $1 - \frac{1}{2^t}$.

So, running the quantum procedure $s + nr$ times with $s \geq 0$ gives a generator set $\{g^1, \ldots, g^{s+nr}\}$ of $N_r(A)^\perp$ with probability at least $1 - \frac{1}{2^s}$ (because $N_r(A)^\perp$ is subgroup of $(A, +)$, which has order p^n). Once that we have such a generating set, Gaussian elimination on the following system of $s + nr$ linear equations

$$\begin{array}{ccccccccc} g_1^1 x_1 & + & g_2^1 x_2 & + & \cdots & + & g_n^1 x_n & \equiv & 0 \mod p \\ \vdots & & \vdots & & \vdots & & \vdots & \vdots & \\ g_1^{s+nr} x_1 & + & g_2^{s+nr} x_2 & + & \cdots & + & g_n^{s+nr} x_n & \equiv & 0 \mod p \end{array} \tag{30}$$

gives a generator set of $N_r(A)$. We have the following main result:

Theorem 5: Given the multiplication table of a n-dimensional nonassociative noncommutative $\mathbb{F}_p$-algebra A, with p a prime number. For each substructure

$$H = N_r(A), N_m(A), N_l(A), N(A), Z(A) \tag{31}$$

of A, there exists a quantum algorithm that using the number of qubits and quantum gates of Table 2, together with a classical post-processing algorithm of complexity $O(n^3)$, finds H with a bounded probability error.

Table 2. Cost in terms of number of qubits and gates of each substructure.

Substructure	N_r, N_m, N_l	N	Z
Number of qubits	$n^3r + nr$	$3n^3r + nr$	$3n^3r + n^2r + nr$
Number of ancillary qubits	$4r + 1$		
Number of gates	$O(n^5r^3)$		

Remark 1: Observe that the number of ancillary qubits is upper-bound by the number of such qubits involved in the quantum oracle. In other steps of the algorithm, ancillary qubits can be reused, either by uncomputation or by measuring and resetting to zero.

5 Examples

We will illustrate the behaviour of algorithm 4.2 with two simple examples. In the first one, we compute the center of an associative 3-dimensional $\mathbb{F}_2$-algebra which is not commutative. In the second one, we obtain the right nucleus of a non associative and non commutative 4-dimensional $\mathbb{F}_2$-algebra. Recall that in our notation n is the dimension of the $\mathbb{F}_2$-algebra, and $r = \lceil \log_2(p) \rceil$ which in this case $p = 2$, so $r = 1$.

Example 1: Consider the following set

$$A = \left\{ \begin{pmatrix} a & b \\ 0 & c \end{pmatrix} \; : \; a, b, c \in \mathbb{F}_2 \right\} \tag{32}$$

with ordinary matrix addition and multiplication. It is a 3-dimensional $\mathbb{F}_2$-algebra associative but not commutative. Let us find $Z(A)$ with our algorithm with respect to the $\mathbb{F}_2$-basis

$$\beta = \left\{ e_1 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, e_2 = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, e_3 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \right\}. \tag{33}$$

The multiplication table of A is

$$\mathbb{M} = \left\{ A_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, A_2 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}, A_3 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \right\}. \tag{34}$$

Because A is associative, f_N is identically zero and we only need to consider the last 3 components of the f_Z function. Each of the commutators in f_Z has 3 coordinates in the β basis, so its expression is

$$(0, x_2, 0, 0, x_1 + x_3, 0, 0, x_2, 0)\,,$$

which can be simplified to $(x_2, x_1 + x_3)$ where $x_1, x_2, x_3 \in \mathbb{F}_2$ by eliminating the coordinates that are always 0. The quantum oracle that performs the unitary operation $U_Z |a\rangle |0\rangle = |a\rangle |f_Z(a)\rangle$ with $a = x_1e_1 + x_2e_2 + x_3e_3$ can be seen in Figure 2. Repeatedly using Algorithm 4.2, we determine that the orthogonal of $Z(A)$ is $\{010, 000, 101\}$ with high probability (in this case, $n = 3$ because the algebra is 3-dimensional and $r = 1$ because we are working

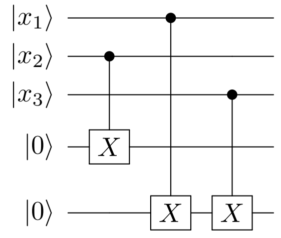


Fig. 2. Oracle for U_Z in Example 1.

over $\mathbb{F}_2$; thus, following the argument after Theorem 4.3, if the number of repetitions is 10, the probability of not obtaining a complete set of generators for $Z(A)^\perp$ is below 1%). This leads to the following system of equations

$$\begin{array}{ccccccccc} 0x_1 & + & 0x_2 & + & 0x_3 & \equiv & 0 & \mod 2 \\ 0x_1 & + & 1x_2 & + & 0x_3 & \equiv & 0 & \mod 2 \\ x_1 & + & 0x_2 & + & x_3 & \equiv & 0 & \mod 2. \end{array} \quad (35)$$

This system has two solutions: $x_1 = 1, x_2 = 0, x_3 = 1$ and $x_1 = 0, x_2 = 0, x_3 = 0$. Therefore, we deduce that $(1, 0, 1)$ generates $Z(A)$. This is, indeed, the correct solution, since the only non-zero element that is mapped to $(0, 0)$ by the hiding function $(x_2, x_1 + x_3)$ is exactly $(1, 0, 1)$.

Example 2: Consider the following multiplication table for a 4-dimensional $\mathbb{F}_2$-algebra:

$$A_1 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad A_2 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} A_3 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} A_4 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

This algebra is neither associative nor commutative. Let us use our algorithm to find its right nucleus. After eliminating zeroes and repeated coordinates, we arrive at a hiding function $f_{N_r(A)}$ whose coordinate expansion is given by $(x_1 + x_2, x_3 + x_4)$, where $x_1, x_2, x_3, x_4 \in \mathbb{F}_2$. The quantum oracle that performs the unitary operation $U_{N_r} |a\rangle |0\rangle = |a\rangle |f_{N_r}(a)\rangle$ with $a = x_1e_1+x_2e_2+x_3e_3+x_4e_4$ can be seen in Figure 3. As in Example 1, repeatedly using Algorithm

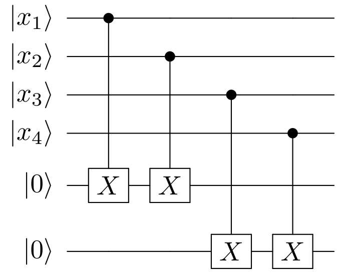


Fig. 3. Oracle for U_{N_r} in Example 2.

4.2, we can obtain (with high probability) that $N_r(A)^\perp$ is generated by $\{(1, 1, 0, 0), (0, 0, 1, 1)\}$.

As a consequence, we can deduce that $\{(1,1,0,0),(0,0,1,1)\}$ generates $N_r(A)$. This is a correct solution, as it is easy to check that these two elements are mapped to $(0,0)$ by the hiding function and that any other element with property can be expressed as a linear combination of them.

6 Conclusions and future work

In this work, we have shown that for a given multiplication table of a n-dimensional nonassociative noncommutative $\mathbb{F}_p$-algebra A, with p a prime number and for each substructure the right, middle, and left nuclei, nucleus and center of A, we can construct an efficient quantum algorithm that computes such a substructure. Our approach is based on the existence of a function that hides the substructure, for which a quantum oracle can be efficiently constructed, both in terms of number of qubits and quantum gates. Such a black box is used by our quantum algorithm, and the concrete number of required qubits and quantum gates can be found in Theorem 4.3. Moreover, we have also proven that solving the corresponding instance of the HSP with just classical means can be very inefficient in some cases. In those situations, we have showed an exponential speed-up with our quantum algorithm.

Our results are based on the existence of efficient quantum solutions to the HSP in the finite Abelian case. However, finding an efficient quantum algorithm (or proving that it is not possible) for the general HSP is still an open problem. In future works, we would like to study the possibility of applying quantum algorithms (as those proposed in [12]) for instances of the HSP that extend the finite Abelian case in order to determine substructures in algebraic structures more general than those considered in this paper.

Acknowledgements

This work has been partially supported by PAPI-21-PF-10, by the Spanish Government, under Project PID2021-123461NB-C22, also in part under grant PID2020-119082RB-C22 funded by MCIN/AEI/ 10.13039/501100011033 and grant AYUD/2021/50994 funded by Gobierno del Principado de Asturias, and by the QUANTUM SPAIN project funded by the Ministry of Economic Affairs and Digital Transformation of the Spanish Government and the European Union through the Recovery, Transformation and Resilience Plan - NextGenerationEU.

References

1. P. W. Shor (1997), *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer*, SIAM J. Comput. 26, 5, pp. 14841509.
2. E. F. Combarro, J. Ranilla and I. F. Ra (2019), *A Quantum Algorithm for the Commutativity of Finite Dimensional Algebras*, IEEE Access vol. 7, pp. 45554-45562.
3. A. A. Albert (1960), *Finite division algebras and finite planes*, Proc. Symp. Appl. Math 10 pp. 5370.
4. I.F. Ra, Elas F. Combarro, J. Ranilla (2009), *Classification of semifields of order 64*, Journal of Algebra, Volume 322, Issue 11, pp. 4011-4029.
5. J.M. Hernndez Cceres, I.F.Ra, *An approach to the Classification of Finite Semifields by Quantum Computing*, Springer Proceedings in Mathematics & Statistics (PROMS, volume 427), pp. 245-260.
6. Lomont, Chris. (2004). *The Hidden Subgroup Problem - Review and Open Problems.* arXiv:quant-ph/0411037

7. Nielsen, M., Chuang, I. (2000) *Quantum Computation and Quantum Information*,10th Anniversary Edition. Cambridge: Cambridge University Press.
8. E. F. Combarro, S. Gonzlez-Castillo (2023), *A Practical Guide to Quantum Machine Learning and Quantum Optimization*. Packt.
9. A. A. Albert (1961), *Generalized twisted fields*, Pac. J. of Math, 11, pp. 1-8.
10. Vlatko Vedral, Adriano Barenco, and Artur Ekert, (1996) *Quantum networks for elementary arithmetic operations*. Phys. Rev. A, 54, pp. 147153
11. Oliveira, David., and Ramos, Rubens. (2007). *Quantum bit string comparator: Circuits and applications*. Quantum Computers and Computing, volume 7
12. Hallgren, Sean., Russell, Alexander., Ta-shma, Amnon. (2002). *The Hidden Subgroup Problem and Quantum Computation Using Group Representations*. SIAM Journal on Computing. 32.