Quantum computing models for algebraic applications

Nikolay Raychev

Abstract - In our days the quantum computers may be used for a great number of algebraic applications exponentially faster in comparison to the classical computing equipment. In this article are considered key aspects of quantum models algebraic applications in terms of the application of the algorithms in a quantum circuit using only elementary quantum operations, which is important for determining the potential applicability of the models in the practice.

Key words: Quantum computing, algebraic applications, operators, gates

1. INTRODUCTION

The quantum computers use the quantum mechanical phenomena such as superposition and quantum entanglement to perform computations. Since the computations are carried out in ways, which the classic computers can not apply for certain tasks such as decomposition of large numbers [1] and simulation of quantum systems [2- 16], the quantum algorithms provide exponential increase of the outputs compared to their classic counterparts. Recently Harrow, Hassidim and Lloyd [17] proposed a quantum algorithm for obtaining certain information for finding $\vec{x}$ of $A\vec{x} = \vec{b}$. First we will set forth the algorithm and some comments about the key aspects relating to the efficient implementation of its quantum circuit. Then we will present an example for a quantum circuit, in order to encourage the experimental efforts for implementing the algorithm.

2. BOOLEAN EXPRESSIONS

Specification: There is an $\boldsymbol{n}$ variables Boolean expression given, the entire possible range of variables should be found within 2^n possible combinations, by which the result of the expression will be **TRUE**.

The Boolean expression consists of the AND-ing of a number of clauses, each clause consists of the OR-ing clauses of K Boolean variables, and each variable may be negated.

There are two well-developed paradigms of Boolean logic.

- The first uses operations of AND, OR and NOT and is called canonical Boolean logic.
- The second uses the operations AND, XOR and NOT and is called Reed-Muller logic (RM)

For constructing quantum circuits for Boolean functions, will be used one ancilla qubit, which initially is initialized to 0 and a Boolean function constructed by CNOT based transformations, which work as follows:

CNOT $(C|t)$ is an operator, whose target qubit is controlled by a set of qubits C, so that $t \notin$ C, the state of t will be changed by $|0\rangle \rightarrow |1\rangle$ or by $|1\rangle \rightarrow |0\rangle$ only and solely if all qubits in C are with state $|1\rangle$; i.e. the new state of the target qubit t will be the result of the operation **disjunction** of the old state of t with the **AND-ing** of the states of the control qubits.

Example 1:

Boolean function: $f(x_0, x_1, x_2) = \overline{x_0} \vee x_1 x_2$

1. Transformation in positive RM polarity : $f = x_0 x_1 x_2 \oplus x_0 \oplus 1$
2. Design of quantum Boolean circuit
   - Initialization of the target qubit with a state $|0\rangle$
   - A CNOT gate is added to each product in this expansion, as the boolean variables in this product are implemented as control qubits,

the result is calculated and stored into the target qubit.

- For the last product which contains only one unit is added a CNOT (*t*)

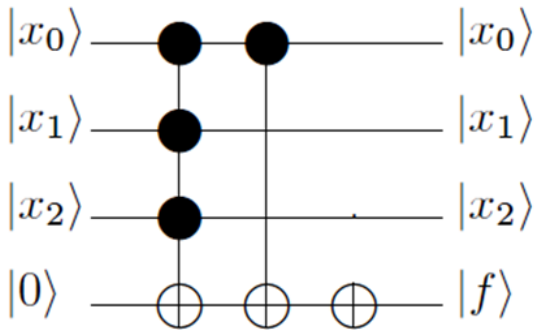


Figure 1 Boolean function: $f(x_0, x_1, x_2) = \overline{x_0} \vee x_1 x_2$

Example 2:

Boolean function $f(x_1, x_2, x_3) = (x_1 \vee x_2 \vee x_3)(\overline{x_1} \vee x_2 \vee x_3)(x_1 \vee \overline{x_2} \vee \overline{x_3})$

1. Transformation into positive RM polarity: $f = x_1 x_2 x_3 \oplus x_2 \oplus x_3$
2. Design of quantum Boolean circuit
   - Initialization of the target qubit with a state $|0\rangle$
   - A CNOT gate is added to each product in this expansion, as the boolean variables in this product are implemented as control qubits, the result is calculated and stored into the target qubit.

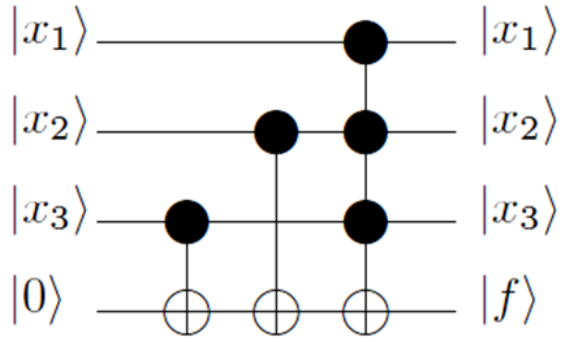


Figure 2: Boolean function $f(x_1, x_2, x_3) = (x_1 \vee x_2 \vee x_3)(\overline{x_1} \vee x_2 \vee x_3)(x_1 \vee \overline{x_2} \vee \overline{x_3})$

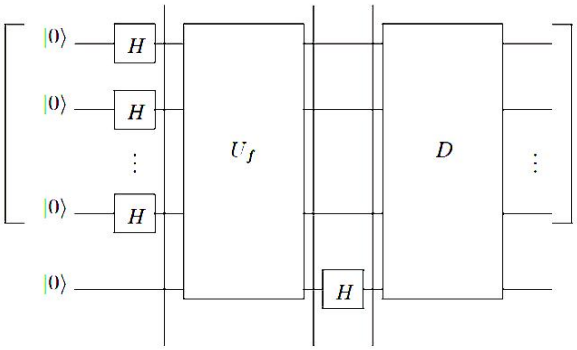


Figure 3 Quantum circuit for solving Boolean expressions

Register preparation. A quantum register with n + 1 qubits is prepared, for all n variables involved in the boolean expression, all qubits are initialized in state $|0\rangle$, the additional qubit will be used as workspace for evaluating the Boolean function

Register initialization. Hadamard gates are applied on the first n qubits, thus was created a superposition of n *qubits and* in parallel are processed all 2^n possible states of the boolean expression

$$|W_1\rangle = (H^{\otimes n} \otimes I)\,|W_0\rangle = \left(\frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} |i\rangle\right) \otimes |0\rangle; N = 2^n$$

Implementation of the Boolean function. An Oracle operator U_f is applied (*by the described procedure in both shown examples*), which in parallel will process all possible states and will retain the outcome in the battery

$$|W_2\rangle = U_f |W_1\rangle = \frac{1}{\sqrt{N}} \left(\sum_{i=0}^{N-1} (|i\rangle \otimes |0 \otimes f(i)\rangle) \right)$$

$$= \frac{1}{\sqrt{N}} \left(\sum_{i=0}^{N-1} (|i\rangle \otimes |f(i)\rangle) \right)$$

Completing the superposition and changing the sign. A Hadamard gate is applied on the working qubit. This will complete the superposition of *n+1 qubits.* The found states satisfying the function will be marked by negative amplitudes.

$$|W_3\rangle = (H^{\otimes n} \otimes I)\,|W_2\rangle$$
$$= \frac{1}{\sqrt{N}}\left(\sum_{i=0}^{N-1}|i\rangle \otimes \left(\frac{|0\rangle + (-1)^{f(i)}|1\rangle}{\sqrt{2}}\right)\right)$$
$$= \frac{1}{\sqrt{P}}\left(\sum_{i=0}^{N-1}(|i\rangle \otimes |0\rangle + (-1)^{f(i)}|1\rangle)\right); \quad P = 2N = 2^{n+1}$$

If M is the number of the possible states of the variables at which the result of the Boolean function will be TRUE (solutions); so that $0 \le M \le N$; ($N = P/2$). $|W_3\rangle$ there will be two possible forms:

1. If in the superposition there is at least one open solution of the Boolean function. $\sum_i(\ldots)'$ is the sum of all i, which are solutions, and $\sum_i(\ldots)''$ is the sum of all i, which are not solutions

$$|W_{3_1}\rangle = \frac{1}{\sqrt{P}}\sum_{i=0}^{N-1}(|i\rangle \otimes (|0\rangle - |1\rangle))' + \frac{1}{\sqrt{P}}\sum_{i=0}^{N-1}(|i\rangle \otimes (|0\rangle + |1\rangle))''$$
$$= \frac{1}{\sqrt{P}}\sum_{i=0}^{N-1}(|i\rangle \otimes |0\rangle)' - \sum_{i=0}^{N-1}(|i\rangle \otimes |1\rangle)' + \frac{1}{\sqrt{P}}\sum_{i=0}^{N-1}(|i\rangle \otimes |0\rangle)'' + \sum_{i=0}^{N-1}(|i\rangle \otimes |1\rangle)''$$

2. If in the superposition is not found a single solution of the Boolean function. The function will have the following form

$$|W_{3_2}\rangle = \frac{1}{\sqrt{P}}\sum_{i=0}^{N-1}(|i\rangle \otimes (|0\rangle + |1\rangle))$$

Application of a Diffusion operator, which increases the amplitudes with a negative sign (the solutions) and decreases the amplitudes with a positive sign

$$D = H^{\otimes n+1}(2|0\rangle\langle 0| - I)H^{\otimes n+1} = 2|\psi\rangle\langle\psi| - I$$

Complexity of the algorithm:
The proposed algorithm consists of a constant number of steps, f:(O(4))
1. Preparing the register for N qubits - O(n).
2. Implementing the Boolean function - O(λ); ($\lambda \le 2^{n-1}$).
3. Completing the superposition and changing the sign - O(1).

4. Diffusion operator - inversion according to the importance - O(4n)

3. ALGEBRAIC EQUATIONS

Lets suppose that an operator A is presented as N × N Hermite matrix with spectral factorization of $A = \sum_j \lambda_j |u_j\rangle\langle u_j|$ (the non-Hermite cases can be explained by some simple modifications of the algorithm, see [17, sec. 4, annex A]). The number of conditioning is defined as $\kappa = max_j|\lambda_j|/min_j|\lambda_j|$. Without the limitations of the community we accept $\kappa^{-1} \le \lambda_j \le 1$ for all j.

The main quantum circuit for the algorithm is shown on fig. 1. The right vector is encoded in the quantum state $\vec{b}$, which expands as $|b\rangle = \sum_j b_j |u_j\rangle$ into a basis of own vectors A. The algorithm starts with the known subroutine for the phase estimation, which involves applying the control unitary unit $U = e^{iAt}$ on $|b\rangle$ for a superposition of different t values. After the phase estimation we obtain a state, which is approximately $\sum_j b_j |\lambda_j\rangle|u_j\rangle$ (Fig. 1). Here $|\lambda_j\rangle$ is a state that encodes an approximation to the eigenvalue λ_j [18, sec. 5].

The next step of the algorithm aims to make the system proportional to $\sum_j b_j \lambda_j^{-1}|u_j\rangle \otimes |Anc.\rangle$. Here $|Anc.\rangle$ is a state of the ancilla qubits. The ancilla qubits decouple from this subset of qubits in state $\sum_j b_j \lambda_j^{-1}|u_j\rangle$ which is proportional to the solution $|x\rangle \propto A^{-1}|b\rangle$. To achieve this transformation is introduced an ancilla qubit, initialized on $|0\rangle$ and are used the states $|\lambda_j\rangle$ after the phase estimation (fig. 1) for performing a controlled Y - rotation $R_y(\theta_j) = e^{-i\theta_j Y/2}$(Y is a Pauli Y operator) of the ancilla qubit such that the state of the system is brought to

$$\sum_j \left(\sqrt{1 - \frac{C^2}{\lambda_j^2}}\,|0\rangle + \frac{C}{\lambda_j}|1\rangle\right) b_j\lambda_j|u_j\rangle \qquad (1)$$

with the rotation angles $\theta_j = 2\arcsin(C/\lambda_j)$. Here the constant $C \leq min_j|\lambda_j| = 0(1/k)$.

Figure 4 Overview of the quantum circuit for solving the linear system $A\vec{x} = \vec{b}$. Each label Anc. represents an ancilla register. Reg. Means any register that stores (intermediate or final) computed results. W is a Walsh-Hadamard transform which applies the Hadamard transform on each qubit of the register. FT represents quantum Fourier transform (FT). The circuit for FT is well known [18]. U_λ is a subroutine, which computes the state $|\tilde{\theta_j}\rangle$ with $\tilde{\theta_j}$, approximating $\theta_j = 2arcsin(C/\lambda_j)$ for the eigenvalues of A, encoded in states $|\lambda_j\rangle$. $U\dagger$ represents the inverse of all the operations before the controlled R_y rotation. For small rotation angles in R_y the final state of the top ancilla bit is $|a_j\rangle$, which approximates $\sqrt{1 - C^2/\lambda_j^2} + C/\lambda_j\ |1\rangle$ with an fidelity up to ε.

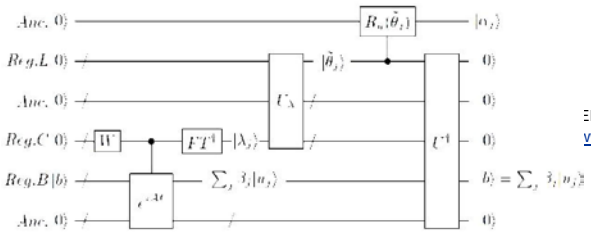
The final step of the algorithm is applying the inversion of a subroutine for phase estimation at the beginning and transforming the register $|\lambda_j\rangle$ back to $|0\rangle$, thus transforming the system to $\sum_j \sqrt{(1 - C^2/\lambda_j^2}\,|0\rangle + (C/\lambda_j)|1\rangle)\beta_j|0\rangle|u_j\rangle$. The projective measurement of the first ancilla qubit, where it is $|1\rangle$, will reduce the final state of Reg. *B* (fig. 1) to the desired state

$$\sum_j C\frac{b_j}{\lambda_j}|u_j\rangle \propto |x\rangle \qquad (2)$$

with probability of $\sum_j |b_j|^2 \cdot |C/\lambda_j|^2$, which is estimated to O ($1/\kappa^2$).

Here we will examine some key aspects of the algorithm related to finding an efficient quantum circuit implementation when using only elementary operations.

A detailed complex analysis in [17] shows that the algorithm runs in O ($\log(N)\kappa 2/\varrho$) time, where ϱ is a total error in the output state $|x\rangle$. It is proven that the complexity, or the cost scaling of the quantum algorithm values as κ and ϱ, is optimal [17, sec. 5, annex A], while in cases such as A, which are symmetrical positively defined, the best classic algorithm of the conjugate gradient has a scaling O ($N\sqrt{\kappa}\log(1/\varrho)$) [19].



Hence the most useful application of the algorithm is limited to situations where neither, nor $1/\varrho$ are large [20].

The major strength of the algorithm is that at under certain conditions (which we will discuss later) it finds the solution $|x\rangle$ with O (logN) value, while any classical algorithm requires at least O(N) operations to record the answer $\vec{x}$. Because the solution is encoded in the quantum state $|x\rangle = \sum_{i=1}^{N} x_i|i\rangle$ and obtaining all the values of x_i still requires operation O(N), the application of the algorithm is further limited to cases where we are only interested in a certain solution that is represented by the expected value ‹x|M|x› for an operator *M*.

At the above restrictions, in order to keep the scaling of the value O(logN), the algorithm assumes that all of the following subroutines are efficient (with scaling of value 0 (poly(logN, κ, 1/ ϵ))).

Preparation of state $|b\rangle$. It is known that in order to prepare an arbitrary quantum state in an N-dimensional Hilbert space are necessary O (poly(N)) elementary gates [18]. This is immediately related to decomposing an arbitrary unitary operation to elementary quantum gates (such as CNOT gates and single-qubit rotations), since preparing any state |b› requires a unitary unit U, such that (for example) the transformation of U|0 · · · 0›= |b› serves as preparing |b' of |b› of |0 · · · 0›, a state that is easy to prepare. There are improved circuits for decomposition of the unitary gates, however, there is no general circuit that will break the poly (N) bound in the scaling of the values. Regardless of this fundamental limitation, there are specific types of states, which can be prepared efficiently.

One important example is the case, considered independently by Zalka [24], Grover and Rudolph [25], Kaye and Mosca [26], where the state |b› = Σ_i b(*i*)|*i*›, corresponding efficiently to an integrable function *b(x)*, can be prepared efficiently. The initial motivation in [24] is to encode a continuous wave function into a quantum state in preparation for quantum simulation. This idea is extended in [27] for

generating a numerous single- and multipart eigenstates for quantum simulation.

While the circuits for state generation in [24 27] are presented in the standard model of a quantum computation gate, Aharonov and Ta-Shma [28, 29] consider the quantum state generation by simulating Hamiltonians of adiabatic evolutions that correspond to slowly varying Markov chains. According to [28, Sec. 4], for a Markov chain described with a matrix M, acting on the distribution of probabilities that in the space of states Ω, for the limiting distribution $\pi = \lim_{t\to\infty} M^t p$ with p, which is an output distribution, if M is row computable and for each $i, j \in \Omega$, $M_{ij}\pi_i = M_{ji}\pi_j$ and π_i/π_j can be efficiently approximated, then the Hamiltonian corresponding to M, defined as $H_M = I - \Lambda M \Lambda^{-1}$, where Λ e is a diagonal operator with $\sqrt{\pi_i}$ at the i^{th} diagonal position, has its own basic state $\Sigma i \sqrt{\pi_i}|i\rangle$. Because we can efficiently simulate an adiabatic evolution starting at a Hamiltonian corresponding to a simple Markov chain and ending at a Hamiltonian corresponding to H_M (See [28, Lemma 2 and 3]), thus the state $\Sigma_i \sqrt{\pi_i}|i\rangle$ can be efficiently prepared.

Hamiltonian simulation e^{-iAt}. The task of the Hamiltonian simulation has been extensively studied. For a basic nondense $N \times N$ Hamiltonian H is shown in [30] that it is not possible to simulate e^{-iHt} into poly (||Ht||, logN) time, which establishes the fundamental limitation for the currently known Hamiltonian simulation schemes. In general, however, it is of greater importance to efficiently simulate sparse Hamiltonian schemes. Especially if H *is* 1 (first) sparse (s- sparse means that each row and column of the Hamiltonian has at most s non-null elements), e^{-iHt} can be filled with O (1) elementary operations [28, 31]. In generally, an s-sparse Hamiltonian can be decomposed efficiently as a sum of O (s^2) 1-sparse Hamiltonians [28, 32]. Since the initial work by Lloyd [2] for cases of time-independent local Hamiltonians, there are several simulation algorithms using formulas [6, 32, 33] and improved, using linear combinations of unitary operators [34], with scaling of the values that is poly (logN, 1/ϱ) and almost linear in ||Ht||.

Another option, algorithmic simulation, using quantum walks [35, 36], has scaling of the values O||Ht||/√ϵ), which is strictly linear in в ||Ht||. None of the presented algorithms so far can not show O (log(1/ϵ)) scaling in 1/ϵ, except the special cases in which H has a specific structure such as being proportional to a discrete Laplacian in any finite dimension [37].

However, on the basis of prior works [32, 36, 38] was recently shown in [39] that in order to simulate e^{-iHt} for an s-sparse Hamiltonian is required only O (s^2||Ht||poly (logN, log(1/ϵ))), breaching the limitation of the previous algorithms for scaling with a view of 1/ϵ. The issue of Hamiltonian simulation in O(log(1/ϵ)) is important, because the classical simulations of the quantum systems suffer from the exponential scaling in n = logN, but they have O (log(1/ϵ)) scaling. The ability to achieve the same O (log(1/ϵ)) scaling in the quantum regime could help answering open questions in numerical analysis [34].

Eigenvalue inversion U_λ. The exact transformation of the state of the system up to (1) requires a non-unitary operation. It is shown n [37] that using quantum circuits to simulate classic subroutines for finding the roots, the state $|\theta_j\rangle$, approximating θ_j up to error ε, can be prepared with O (polylog(1/ε)) value.

Example: solving a 2 × 2 system

Here we present a 4-qubit quantum circuit that solves the smallest significant example of the task: system 2 × 2. The purpose of the example is to illustrate the algorithm and the potential experimental implementation through the current available methods. Hence the simplification with respect to the general quantum circuit implementation discussed in the previous section is possible. In the example $|b\rangle$ is a one-qubit state, which is easy to prepare. The quantum circuit for realizing Hamiltonian simulation e^{-iAt} with elementary operations is found via a heuristic algorithm developed in some previous works [40, 41]. There is no guarantee for the efficiency of the heuristics for large matrices but for our purpose in this example they are sufficient. In order to simplify the eigenvalue inversion subroutine U_λ (Fig. 1), A

is chosen such that it has eigenvalues that are powers of 2, so that the phase estimation subroutine will generate states that exactly encode the eigenvalues, making it simple to find their reciprocals.

As we showed before in the general case, the representation from $\Sigma_j \beta_j |\lambda_j\rangle|u_j\rangle$ to $\Sigma j \beta_j \lambda^{-1}{}_j |\lambda_j\rangle|u_j\rangle \propto |x\rangle$ should in principle use controlled Y rotation with angle $\theta j = 2\arcsin(C/\lambda_j)$. Here we define C such that the small-angle approximation $\arcsin(C/\lambda_j) \approx C/\lambda_j$ to be retained. Although the inversion scheme presented in the example is purely *ad hoc* (special to the specific case - translator's note), with additional qubits in Reg.C (Fig. 1) the implementation of U_λ, as described in [37] is possible.

$$A = \frac{1}{2}\begin{pmatrix} 3 & 1 \\ 1 & 3 \end{pmatrix}; \vec{b} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \qquad (3).$$

The circuit for solving the linear system is shown in Fig. 2. Assuming that $|b_1|^2 + |b_2|^2 = 1$, the vector $\vec{b}$ can be encoded in the state $|b\rangle = b_1|0\rangle + b_2|1\rangle$. The eigenvalues of A are $\lambda_1 = 1$ и $\lambda_2 = 2$ with corresponding eigenvectors $|u_1\rangle$ and $|u_2\rangle$. Note that λ_1 and λ_2 can be accurately encoded by $|x_2x_3\rangle = |01\rangle$ and $|x_2x_3\rangle = |10\rangle$ respectively. Therefore, after the phase estimation stage of the 3-qubit system $|x_2x_3x_4\rangle$ reads $\beta_1|01\rangle|u_1\rangle + \beta_2|10\rangle|u_2\rangle$, where β_1 and β_2 are coefficients from the decomposition of $|b\rangle$in the basis of eigenvectors of *A*.

To obtain the state $|\theta_j\rangle$ for the inversion of the eigenvalues (Fig. 5), we use a method special for the case, which does not require any ancilla qubits. We first apply a SWAP gate between $|x_2\rangle$ and $|x_3\rangle$, so that the 3-qubit system $|x_2x_3x_4\rangle$ is transformed to the state $\beta_1|10\rangle|u_1\rangle + \beta_2|01\rangle|u_2\rangle$. We can now interpret $|x_2x_3\rangle = |10\rangle$ as a state, encoding the inverse eigenvalue $2\lambda^{-1}1 = 2$ and $|x_2x_3\rangle = |01\rangle$ as that encoding $2\lambda^{-1}{}_2 = 1$. In other words, after the SWAP gate following the phase estimation (Fig. 2), the state $|x_2x_3x_4\rangle$ becomes $\Sigma\, {}^2{}_{j=1}\, \beta_j |2\lambda^{-1}{}_j\rangle|u_j\rangle$.

Then we use the states $|2\lambda^{-1}{}_j\rangle$ in $|x_2x_3\rangle$ as a control register to execute a Y rotation $R_y(\tilde{\theta}_j)$ on qubit $|x_1\rangle$ with $\tilde{\theta}_j = 2^{1-r}\pi / \lambda_j = 2C / \lambda_j$ which approximates $\theta_j = 2\arcsin(C / \lambda_j)$. We have previously assumed that $C \le \min_j |\lambda_j|$. Hence we allow $r \ge \log_2(2\pi) \approx 2.65$. In general *r* cannot be too small otherwise the approximation with small angles $\tilde{\theta}_j$ of θ_j will become invalid. At the same time *r* can not be too large, because at the larger *r is* less likely to be obtained a solution and also the more acute angles will have to be solved in control rotation gates, which places greater challenges upon the implementation. Suppose the minimum angle realizable is ω, then $r \le \log2(\pi/\omega)$.

Phase estimation

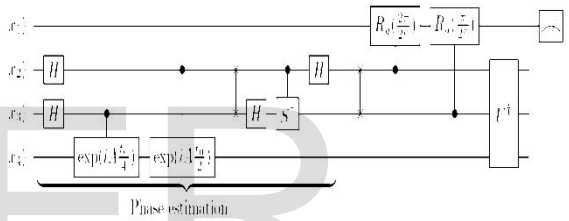


Figure 5 The example quantum circuit for solving the 2×2 linear system $A\vec{x} = \vec{b}$. Here $|x_1\rangle$, $|x_2x_3\rangle$ u $|x_4\rangle$ correspond respectively to the top ancilla qubit, register C and register B in Fig. 1. We assume $t_0 = 2\pi$ and that r is a parameter that varies between $\log_2(2\pi)$ and $\log_2(\pi/\omega)$ with ω minimum angle that can be resolved. Here U† represents an inversion of all operations before the R_y rotations. $r > 0$ is a parameter that will determine the probability of the final state. Initially $|x_4\rangle = |b\rangle$ and $|x_1\rangle, |x_2\rangle$ u $|x_3\rangle$ are all $|0\rangle$.

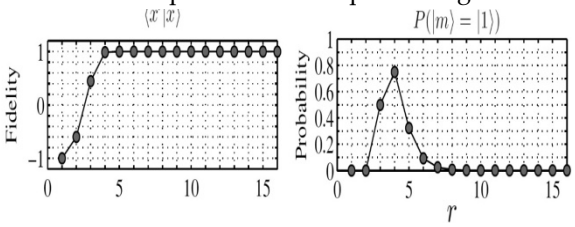
The numerical results simulating the circuit with different values of *r*, are shown in Fig. 7. When the value of *r* is sufficiently large, the fidelity $\langle x'|x\rangle$ of the solution approximates to 1. Here $|x'\rangle$ is a state of $|x_3x_4\rangle$, where $|x_1\rangle$ is measured to be $|1\rangle$ in the final state. From the previous analysis we see that $|x'\rangle = \cos\tilde{\theta}_1|u_1\rangle + \sin\tilde{\theta}_2|u_2\rangle$. $|x\rangle$ corresponds to the analytical solution $\vec{x} = (3/8, -1/8)^T$. The numerical results in Fig. 3 also show that since *r* increases beyond a certain point ($r \approx 4$), the probability of measuring the ancilla bit as $|1\rangle$ drops, which indicates that, since *r* increases, the solutions obtained in the

final state in a register *b*, become more accurate, although less likely to obtain.

Figure 6 A numerical calculation of the fidelity ‹x'|x› of the quantum solution and the probability of obtaining |x'› as a function of r. |m› e is the state |x₁› after measurement (Fig. 2).

4. CONCLUSION

In conclusion lets note that, in this work we have discussed the general and special cases of applying the quantum models for algebraic applications. The quantum computers may be used to solve different normal algebraic equations and Boolean expressions. The proposed algorithmic models have a huge range of possible applications, because large systems of algebraic expressions are used anywhere in the field of science and technology. There is one interesting open question whether the scaling can be improved, since these results are for constant coefficients, because this allows for analytical analysis of the errors. This approach can be used to resolve algebraic equations with time-dependent coefficients, due to which the analysis for errors will be more difficult. Our results may motivate the experimenters to consider and perform basic quantum gates for



applying the algorithm and verify the results of it.

REFFERENCES

[1] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, 1st ed. (Cambridge University Press, Cambridge, UK, 2000).
[2] P. W. Shor, SIAM Journal on Computing 26, 1484 (1997).
[3] L. K. Grover, Physical Review Letters 79, 325 (1997).
[4] C. H. Bennett and G. Brassard, in *Proceedings of IEEE international Conference on Computers, Systems and Signal Processing, Bangalore, India* (IEEE Press, New York, 1984), p. 175.
[5] A. K. Ekert, Phys. Rev. Lett. 67, 661 (1991).
[6] C. Elliott, New Journal of Physics 4, 46 (2002). 12
[7] C. Elliott, D. Pearson, and G. Troxel, in *Proceedings of the ACM SIGCOMM 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, August 25-29, 2003, Karlsruhe, Germany.* (PUBLISHER, ADDRESS, 2003), pp. 227–238.
[8] C. Elliott, IEEE Security & Privacy 2, 57 (2004).
[9] C. Elliott *et al.*, in *Current status of the DARPA quantum network (Invited Paper)*, edited by E. J. Donkor, A. R. Pirich, and H. E. Brandt (SPIE, ADDRESS, 2005), No. 1, pp. 138–149.
[10] J. H. Shapiro, New Journal of Physics 4, 47 (2002).
[11] B. Yen and J. H. Shapiro, IEEE Journal of Selected Topics in Quantum Electronics 9, 1483 (2003).
[12] S. Lloyd *et al.*, SIGCOMM Comput. Commun. Rev. 34, 9 (2004).
[13] I.-M. Tsai and S.-Y. Kuo, IEEE Transactions on Nanotechnology 1, 154 (2002).
[14] S.-T. Cheng and C.-Y. Wang, IEEE Transactions on Circuits and Systems I: Regular Papers 53, 316 (2006).
[15] J. C. Garcia-Escartin and P. Chamorro-Posada, Phys. Rev. Lett. 97, 110502 (2006).
[16] M. Oskin, F. T. Chong, and I. L. Chuang, Computer 35, 79 (2002).
[17] D. Copsey *et al.*, IEEE Journal of Selected Topics in Quantum Electronics 9, 1552 (2003).
[18] C. H. Bennett and S. J. Wiesner, Physical Review Letters 69, 2881 (1992).
[19] X. S. Liu, G. L. Long, D. M. Tong, and F. Li, Phys. Rev. A 65, 022304 (2002).
[20] A. Grudka and A. W´ojcik, Phys. Rev. A 66, 014301 (2002).
[21] C.-B. Fu *et al.*, JOURNAL OF THE KOREAN PHYSICAL SOCIETY 48, 888891 (2006).
[22] A. Winter, IEEE Transactions on Information Theory 47, 3059 (2001).
[23] H. Concha, J.I.; Poor, IEEE Transactions on Information Theory 50, 725 (2004).
[24] M. Fujiwara, M. Takeoka, J. Mizuno, and M. Sasaki, Physical Review Letters 90, 167906 (2003).
[25] J. R. Buck, S. J. van Enk, and C. A. Fuchs, Phys. Rev. A 61, 032309 (2000).
[26] M. Huang, Y. Zhang, and G. Hou, Phys. Rev. A 62, 052106 (2000).
[27] B. J. Yen and J. H. Shapiro, in *Two Problems in Multiple Access Quantum Communication*, edited by S. M. Barnett *et al.* (AIP, ADDRESS, 2004), No. 1, pp. 25–28.

[28] B. J. Yen and J. H. Shapiro, Physical Review A (Atomic, Molecular, and Optical Physics) 72, 062312 (2005).
[29] B. Sklar, IEEE Communications Magazine 21, 6 (1983).
[30] B. Sklar, *Digital Communications*, 2nd ed. (Prentice Hall, Upper Saddle River, New Jersey 07458, 2000).
[31] P. D. Townsend, Nature 385, 47 (1997).
[32] V. Fernandez *et al.*, in *Quantum key distribution in a multi-user network at gigahertz clock rates*, edited by G. Badenes, D. Abbott, and A. Serpenguzel (SPIE, ADDRESS, 2005), No. 1, pp. 720–727.
[33] Nikolay Raychev. Dynamic simulation of quantum stochastic walk. In International jubilee congress (TU), 2012.
[34] Nikolay Raychev. Classical simulation of quantum algorithms. In International jubilee congress (TU), 2012.
[35] Nikolay Raychev. Interactive environment for implementation and simulation of quantum algorithms. CompSysTech'15, DOI: 10.13140/RG.2.1.2984.3362, 2015
[36] Nikolay Raychev. Unitary combinations of formalized classes in qubit space. International Journal of Scientific and Engineering Research 04/2015; 6(4):395-398. DOI: 10.14299/ijser.2015.04.003, 2015.
[37] Nikolay Raychev. Functional composition of quantum functions. International Journal of Scientific and Engineering Research 04/2015; 6(4):413-415. DOI:10.14299/ijser.2015.04.004, 2015.
[38] Nikolay Raychev. Logical sets of quantum operators. International Journal of Scientific and Engineering Research 04/2015; 6(4):391-394. DOI:10.14299/ijser.2015.04.002, 2015.
[39] Nikolay Raychev. Controlled formalized operators. In International Journal of Scientific and Engineering Research 05/2015; 6(5):1467-1469, 2015.
[40] Nikolay Raychev. Controlled formalized operators with multiple control bits. In International Journal of Scientific and Engineering Research 05/2015; 6(5):1470-1473, 2015.
[41] Nikolay Raychev. Connecting sets of formalized operators. In International Journal of Scientific and Engineering Research 05/2015; 6(5):1474-1476, 2015.
[42] Nikolay Raychev. Indexed formalized operators for n-bit circuits. International Journal of Scientific and Engineering Research 05/2015; 6(5):1477-1480, 2015.
[43] Nikolay Raychev. Converting the transitions between quantum gates into rotations. International Journal of Scientific and Engineering Research 06/2015; 6(6): 1352-1354. DOI:10.14299/ijser.2015.06.001, 2015.
[44] Nikolay Raychev. Quantum algorithm for non-local coordination. International Journal of Scientific and Engineering Research 06/2015; 6(6):1360-1364. DOI:10.14299/ijser.2015.06.003, 2015.
[45] Nikolay Raychev. Universal quantum operators. International Journal of Scientific and Engineering Research 06/2015; 6(6):1369-1371. DOI:10.14299/ijser.2015.06.005, 2015.
[46] Nikolay Raychev. Ensuring a spare quantum traffic. International Journal of Scientific and Engineering Research 06/2015; 6(6):1355-1359. DOI:10.14299/ijser.2015.06.002, 2015.
[47] Nikolay Raychev. Quantum circuit for spatial optimization. International Journal of Scientific and Engineering Research 06/2015; 6(6):1365-1368. DOI:10.14299/ijser.2015.06.004, 2015.
[48] Nikolay Raychev. Encoding and decoding of additional logic in the phase space of all operators. International Journal of Scientific and Engineering Research 07/2015; 6(7): 1356-1366. DOI:10.14299/ijser.2015.07.003, 2015.
[49] Nikolay Raychev. Measure of entanglement by Singular Value decomposition. International Journal of Scientific and Engineering Research 07/2015; 6(7): 1350-1355. DOI:10.14299/ijser.2015.07.004, 2015.
[50] Nikolay Raychev. Quantum algorithm for spectral diffraction of probability distributions. International Journal of Scientific and Engineering Research 08/2015; 6(7): 1346--1349. DOI:10.14299/ijser.2015.07.005, 2015.
[51] Nikolay Raychev. Reply to "The classical-quantum boundary for correlations: Discord and related measures". Abstract and Applied Analysis 11/2014; 94(4): 1455-1465, 2015.
[52] Nikolay Raychev. Reply to "Flexible flow shop scheduling: optimum, heuristics and artificial intelligence solutions". Expert Systems; 25(12): 98-105, 2015.
[53] Nikolay Raychev. Classical cryptography in quantum context. Proceedings of the IEEE 10