Quantum Gauss-Jordan Elimination and Simulation of Accounting Principles on Quantum Computers

Do Ngoc Diep $^{1,4}$ · Do Hoang Giang $^2$ · Nguyen Van Minh $^3$

Abstract The paper is devoted to a version of Quantum Gauss-Jordan Elimination and its applications. In the first part, we construct the Quantum Gauss-Jordan Elimination (QGJE) Algorithm and estimate the complexity of computation of Reduced Row Echelon Form (RREF) of $N \times N$ matrices. The main result asserts that QGJE has computation time is of order $2^{N/2}$. The second part is devoted to a new idea of simulation of accounting by quantum computing. We first expose the actual accounting principles in a pure mathematics language. Then, we simulate the accounting principles on quantum computers. We show that, all accounting actions are exhausted by the described basic actions. The main problems of accounting are reduced to some system of linear equations in the economic model of Leontief. In this simulation, we use our constructed Quantum Gauss-Jordan Elimination to solve the problems and the complexity of quantum computing is a square root order faster than the complexity in classical computing.

The authors’ thanks are addressed to VIASM for a financial support for a scientific stay of the first author at that Institution.

Do Ngoc Diep
dndiep@math.ac.vn

Do Hoang Giang
dhgiang84@gmail.com

Nguyễn Văn Minh
nvminh07@gmail.com

1 Institute of Mathematics, Vietnam Academy of Sciences and Technology, 18 Hoàng Quoc Viet road, Cầu Giấy District, 10307, Hanoi, Vietnam

2 K47A1T, Department of Mathematics, Mechanics and Informatics, College of Natural Sciences, Vietnam National University, 40 Nguyễn Trãi road, Thanh Xuân District, Hanoi, Vietnam

3 Department of Mathematics, Thương Tin High School, Trần Phú Road, Thượng Thịnh town, Thượng Thịnh District, Hanoi, Vietnam

4 Thang Long University, Nghiêm Xuan Yem road, Hoàng Mai district, Hanoi, Vietnam
**Keywords** Quantum Gauss-Jordan Elimination · Complexity · Quantum algorithm · Accounting principles · Quantum search · System of linear equations in the economic model of Leontief

1 Introduction

First, in this introduction, we briefly report of the problem of Gauss-Jordan Elimination procedure and simulation of accounting principles on quantum computers.

**Gauss-Jordan Elimination** Let us consider a general system of linear equations

\[
\begin{align*}
    a_{11}x_1 + \cdots + a_{1n}x_n &= b_1 \\
    \vdots & \quad \vdots \\
    a_{m1}x_1 + \cdots + a_{mn}x_n &= b_m
\end{align*}
\]

As usually denote

\[
    A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\
                       \vdots & \ddots & \vdots \\
                       a_{m1} & \cdots & a_{mn} \end{bmatrix},
    \quad x = \begin{bmatrix} x_1 \\
                           \vdots \\
                           x_n \end{bmatrix},
    \quad b = \begin{bmatrix} b_1 \\
                           \vdots \\
                           b_m \end{bmatrix},
\]

we have the system in matrix form

\[ Ax = b, \]

with the augmented matrix \( \tilde{A} = [A|b] \).

It is well-known from linear algebra that, if the system is consistent, by using elementary transformations, we can reduce the augmented matrix of the system to the so called **Reduced Row Echelon Form (RREF)** and after re-enumerate the variables, we can rewrite it in the form

\[
    RREF ([A|b]) = \begin{bmatrix}
    1 & \cdots & 0 & \tilde{a}_{1, r+1} & \cdots & \tilde{a}_{1, n} & \tilde{b}_1 \\
    \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\
    0 & \cdots & 1 & \tilde{a}_{r, r+1} & \cdots & \tilde{a}_{r, n} & \tilde{b}_r \\
    0 & \cdots & 0 & 0 & \cdots & 0 & 0 \\
    \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots
    \end{bmatrix}
\]

The solutions of the system are on an affine \( m \)-plane, where \( m = n - r \), \( r = rank(\tilde{A}) = rank(A) \) (following Kronecker’s Theorem) with a vector basis

\[
\begin{bmatrix}
    -\tilde{a}_{1, r+1} \\
    \vdots \\
    -\tilde{a}_{r, r+1} \\
    1 \\
    0 \\
    \vdots \\
    0
\end{bmatrix},
\begin{bmatrix}
    -\tilde{a}_{1, r+2} \\
    \vdots \\
    -\tilde{a}_{r, r+2} \\
    0 \\
    1 \\
    \vdots \\
    0
\end{bmatrix},
\begin{bmatrix}
    \cdots \\
    \vdots \\
    \cdots \\
    0 \\
    0 \\
    \vdots \\
    1
\end{bmatrix}
\]
and an affine point

\[
\begin{bmatrix}
\tilde{b}_1 \\
\vdots \\
\tilde{b}_r \\
0 \\
0 \\
0
\end{bmatrix}.
\]

The solutions are in the form

\[
\begin{cases}
x_1 = -\tilde{a}_{1,r+1}x_{r+1} - \cdots - \tilde{a}_{1,n}x_n + \tilde{b}_1 \\
\vdots \\
x_r = -\tilde{a}_{r,r+1}x_{r+1} - \cdots - \tilde{a}_{r,n}x_n + \tilde{b}_r \\
x_{r+1} = x_{r+1} \\
x_{r+2} = x_{r+2} \\
\vdots \\
x_n = x_n 
\end{cases}
\]

In the particular case, where \( r = n = N \), the system has an unique solution. Starting from here, we consider the only this nondegenerate case. The procedure of producing the RREF of the augmented matrix \( \tilde{A} = [A|b] \) is written as some computer program. We refer the readers to [2] for a program in C language. It is easy to show that, this computation needs at least the time of order \( 2^N \). The most complexity time is paid to find a pivotal element.

In this paper, we provide this procedure on quantum computers, and show that, we could have the complexity time of order \( 2^{N/2} \).

As application, we observe that, in accounting, all recording accounts and business actions can be interpreted as vectors and linear operations in linear algebra. We construct in this paper a linear algebra model for accounting. In particular, we define accounts as some 2-component vectors (one component for debit and one for credit). The problems of accounting are simulated on computer programs like Money in Excel, etc. The next essential observation is that, we could simulate accounting by quantum computing and therefore, we use the new technology of quantum computers.

The paper is organized as follows. We state our QGJE Algorithm and the Main Theorem in §2. In §3, we prepare the main background of quantum computing, and in §4, the main theorem is proved. In §5, we describe accounting as a pure mathematical theory, and finally in §6, we show how one can simulate accounting on quantum computers and we have the solution to the economic problem in Leontief modelling.

## 2 Quantum Gauss-Jordan Elimination Algorithm

This section is devoted to describe the algorithm and the statement of the quantum Gauss-Jordan Elimination Algorithm.

The quantum Gauss-Jordan Elimination Algorithm can be described as follows. Suppose we have a matrix \( \tilde{A} = [A|b] \), for which we want to find the \( RREF(\tilde{A}) \). The algorithm can be described as follows.
Step 0: Set $i = 0$, $j = 1$.
Step 1: Change $i$ to $i + 1$, and if $i > N$, exit.
Step 2: Use the Grover’s quantum search algorithm to find out one non-zero entry, in the $i^{th}$ column from $a_{ii}$ to $a_{Ni}$.
Step 3: If the search is successful, we can assume that, the non-zero entry is $a_{ji}$, we change $i^{th}$ row and $j^{th}$ row, and then produce the $i^{th}$ leading 1 in the $i^{th}$ place as $a_{ii}$, else back to the Step 1.
Step 4: Eliminate all the entries $a_{1i}, \ldots, a_{i-1i}, a_{i+1i}, \ldots, a_{Ni}$ in the $i^{th}$ column.
Step 5: Change $j$ to $j + 1$ and if $j > N$ exit, else go back to Step 1.

Remark that, in the procedure, the most complexity time is paid to the problem of finding first non-zero element to provide the pivotal element. We propose to use here the Grover’s search code and our main result is as follows.

**Theorem 2.1** (Main Theorem) *In the Quantum Gauss-Jordan Elimination Algorithm, one needs at most*

$$4N + N(N + 1)^2 + \sqrt{2}(\sqrt{2})^N - 1 \sim O(2^{N/2})$$

*operations.*

Proof of this theorem is the main goal of the Section 4.

3 Quantum Grover’s Search Algorithm and some Background of Quantum Computing

Before prove the main theorem, we review briefly some notions from quantum computing.

3.1 Quantum Gates

Let us recall some fundamental gates in quantum computing, for a detailed exposition, see [3, 4].

**Definition 3.1 (Hadamard gate)** The 1-qubit Hadamard gate $H$ is defined by the matrix

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

The Hadamard gate acts on 1-qubits as follows

$$|x\rangle \xrightarrow{H} \frac{1}{\sqrt{2}}((-1)^x|x\rangle + |1 - x\rangle).$$

**Definition 3.2 (Phase gate)** The phase 1-qubit gate $\Phi$ is defined by the matrix

$$\Phi = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/2} \end{bmatrix}.$$

It produces an action

$$|x\rangle \xrightarrow{\Phi} \frac{1}{\sqrt{2}}e^{i\pi/2}|x\rangle.$$
**Definition 3.3 (Controlled-NOT (CNOT) gate)** The 2-qubit CNOT (XOR) gate is defined by the matrix
\[
C = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}.
\]
The action of this gate is given by
\[
|x\rangle \mapsto |x\rangle |x \oplus y\rangle,
\]
where \(x \oplus y := (x + y) \mod 2\).

**Definition 3.4 (Unitary gate)** An unitary \(n\)-qubit gate is defined by an unitary matrix \(U\). It produces an action
\[
|x_1 \cdots x_n\rangle \mapsto U|x_1 \cdots x_n\rangle.
\]
In general, one unitary operator acts non-trivially on \(n\)-qubits.

In classical computing, one implemented complicated operations as a sequence of simple operations. In quantum computing, we do the same thing, but the complexity is significantly reduced.

The goal is to choose some finite set of gates and by constructing a circuit using only gates from that set, we can implement interesting quantum protocols.

The following result is well-known, see [4].

**Theorem 3.1 (Universal gates)** The Hadamard gate \(H\), the CNOT gate and the phase gate \(\Phi\) generate all the other gates.

**Definition 3.5 (The Black-Box model (Oracle))** Given an unknown function \(f\) of type
\[
f : \{0, 1\}^n \rightarrow \{0, 1\},
\]
we define the quantum operation
\[
U_f : |x\rangle |y\rangle \mapsto |x\rangle |y \oplus f(x)\rangle,
\]
and it is so called black-box, that computes the function \(f\).

The black-box can be illustrated as the following
\[
|x\rangle |y\rangle \xrightarrow{f} |x\rangle |y \oplus f(x)\rangle.
\]

### 3.2 Quantum Grover’s Search Algorithm

Given a black-box, that implements computing a Boolean function of type
\[
f : \{0, 1\}^n \rightarrow \{0, 1\}.
\]

#### 3.2.1 Grover’s Search Original Problem

Given a Boolean function \(f : \{0, 1\}^n \rightarrow \{0, 1\}\), defined by \(f_k(x) = \delta_{x k}\). Find \(k\).
3.2.2 Quantum Grover’s Search Algorithm

One supposed to use the following Scheme:

\[-f_k - H - f_0 - H - \]

This algorithm has special effect (see [3, 4]): Choose the state

\[|\psi\rangle = \frac{1}{\sqrt{2^N/2}} \sum_{i=0}^{N-1} |i\rangle = H|0\rangle,\]

and produces the Grover’s iterate

\[|\psi\rangle(|0\rangle - |1\rangle) \xrightarrow{-f_k - H - f_0 - H -} \text{measurement}(|0\rangle - |1\rangle).\]

This iterate keeps unchanged any basic element \(|i\rangle\) except for \(|k\rangle\) which is reflecting to \(-|k\rangle\), i.e. a reflection in the hyperplane perpendicular to the plane

\[H|0\rangle = |\psi\rangle = \frac{1}{\sqrt{2^N}} \sum_{i=0}^{N-1} |i\rangle,\]

It is a rotation in the plane generate by \(|k\rangle\) and \(|\psi\rangle\), an the Grover’s iterate is a rotation of twice the angle from \(|k\rangle\perp\) to \(|\psi\rangle\perp\). Therefore it is a rotation in the plane generated by \(|k\rangle\) and \(|\psi\rangle\), and the desired state \(|k\rangle\) after \(m\) repeated rotations is exactly \(\Phi\), the iterate should be repeated \(m\) times, such that

\[(2m + 1)\Phi \approx \frac{\pi}{2} \quad \text{or} \quad m \approx \frac{\pi}{2\Phi} - \frac{1}{2},\]

since

\[\frac{1}{2^{N/2}} = \sin \Phi \approx \Phi,\]

we have \(m \approx \frac{\pi}{4} 2^{N/2}\).

This estimate is optimal in the sense that, any another quantum algorithm for seaching an unstructured database must take time of order \(O(2^{N/2})\).

4 Proof of the Main Theorem

Les us come back to the proof of the main theorem. Initialize \(i = 0, j = 1\).

Step 1. Change \(i\) to \(i + 1\), we need one addition and control for whether the value \(i > N\), we need one CNOT gate.

Step 2. We consider the latter of \(i^{th}\) column \(\begin{bmatrix} a_{ii} \\ \vdots \\ a_{Ni} \end{bmatrix}\). Each \(a_{ji}\) is either zero or non-zero and we search for a nonzero value.
In classical computation, one needs to check all possible values of entries in the column and one needs $2^n$ operations, where $n = N - i + 1$. In Quantum Grover’s Search algorithm, we do to proceed $2^{n/2}$ operations.

Suppose we have found a non-zero element and denote it by $a_{ji}$.

**Step 3.** To interchange $i^{th}$ row and $j^{th}$ row, we need $3n$ operations as following

$$t := a_{ik}, \quad a_{ik} := a_{jk}, \quad a_{jk} := t,$$

and produce the $i^{th}$ modified row

$$r_i = [0, \ldots, 0, 1, \frac{a_{i1} + 1}{a_{ii}}, \ldots, \frac{a_{iN}}{a_{ii}}].$$

For this, we need $n$ multiplications with $\frac{1}{a_{ii}}$.

Totally in this step, we need $4n$ arithmetic operations.

**Step 4.** Eliminate all other entries $a_{11}, \ldots, a_{i-1,1}, a_{i+1,1}, \ldots, a_{Ni}$ in the $i^{th}$ column.

For eliminating each $a_{ji}$ we need $n$ multiplications $i^{th}$ row by $a_{ji}$ and $n$ subtractions.

And we repeat $N - 1$ times.

Totally in this step, we need $2n(N - 1)$ arithmetic operations.

**Step 5.** Change $j$ to $j + 1$, we need one addition and control for the value $j > N$, we need one CNOT gate.

**All Step.** Totally in all step, we need at most

$$\sum_{n=1}^{N} \left(2^{n/2} + 4n + 2n(N - 1) + 1 + 1 + 1 + 1\right)$$

operations.

The following lemma is a trivial combinatorics.

**Lemma 4.1 (Combinatorial Lemma)**

1,\hspace{1cm} \sum_{n=1}^{N} n = \frac{N(N + 1)}{2},

2,\hspace{1cm} \sum_{n=1}^{N} 2^{n/2} = \sqrt{2} \left(\sqrt{2}\right)^N - 1.

*Finish the proof of the theorem.* Denote the integral part of a real number $a$ by $[a]$. 
Finally, we have
\[
\sum_{n=1}^{N} \left( \frac{2^n}{2} + 2n(N - 1) + 4n + 1 + 1 + 1 + 1 \right) = \\
\sum_{n=1}^{N} \left( \frac{2^n}{2} + 2n(N - 1 + 2) + 4 \right) \\
\sum_{n=1}^{N} \left( \frac{2^n}{2} + 2n(N + 1) + 4 \right) \\
4 + 2(N + 1) \left[ \sum_{n=1}^{N} n \right] + \left[ \sum_{n=1}^{N} \frac{2^n}{2} \right] \\
4N + 2(N + 1) \frac{N(N + 1)}{2} + \left[ \sqrt{2} \left( \frac{\sqrt{2}}{2} - 1 \right) \right] \\
4N + N(N + 1)^2 + \left[ \sqrt{2} \left( \frac{\sqrt{2}}{2} - 1 \right) \right] \\
\approx O(2^{N/2}).
\]

\[\square\]

**Corollary 4.1** For \( N \gg 1 \), the number of operations needed to produce the RREF of a square matrix using QGJE algorithm is of order \( O(2^{N/2}) \).

### 5 Accounting Principles in Pure Mathematical Theory

This section is devoted to a new idea of simulation of accounting by quantum computing theory. We first expose the actual accounting principles in a purely mathematical language. After that, we simulate the accounting principles on quantum computers. We show that, all accounting actions are exhausted by the described basic actions. The main problems of accounting are reduced to some system of linear equations in the economic model of Leontief. In this simulation, we use our constructed *Quantum Gauss-Jordan Elimination* to solve the problems and the complexity of quantum computing, which is a square root order faster than the complexity in classical computing.

#### 5.1 Motivation

Let us remind the general about accounting. It is clear that, accounting is the language of business. More precisely, *accounting is an information system that identifies, records, and communicates the economic events of an organization to interested users. The main economic events are recorded and then processed by the own rules of accounting in order to give adequate informations to managers and users.*

There is a basic accounting equation
\[
\text{Assets} = \text{Liabilities} + \text{Owner’s Equity},
\]

or the extended basic accounting equation
\[
\text{Assets} = \text{Liabilities} + \text{Owner’s Equity}.
\]
5.2 Mathematical Accounting

It is well-known that, accounting is the information system that identifies, records, and communicates the economic events of an organization to interested users. In some ordinary language (see e.g. [5]), people record these events by transaction, i.e. the economic events of an enterprise that are recorded by accountants. Accountants then journalize the transaction into some account, normally, in form of a T-Account is a quantity, that has some name for reference, some amount (number) that should be putted in some debit or credit side. The amount is normally some amount of money or equivalents, that we can add some of them together or multiply with some scalar. This notion can be expressed in a pure mathematical language as follows.

**Definition 5.1** A T-account \([x|y]\) is a real 2-dimensional vector space with a fixed standard basis consisting of two vectors: \([0] := [1|0]\) for debit side and \([1] := [0|1]\) for credit side. Any vector of form \(x[0] + y[1]\) represents an amount \(x\) on debit side and an amount \(y\) on the credit side.

One transaction can be journalized as a subspace in a sum of T-accounts, corresponding to that action.

**Corollary 5.1** A transaction can be presented as a subspace in the vector space, generated by the corresponding T-accounts.

**Corollary 5.2** A journal can be presented as the vector space, generated by the corresponding T-accounts, i.e. a vector space of form
\[
(\mathbb{R}^2)^k = \bigoplus_k (\mathbb{R}^2) := \mathbb{R}^2 \oplus \cdots \oplus \mathbb{R}^2.
\]

**Corollary 5.3** A special journal can be presented as the vector space, generated by the corresponding T-accounts, i.e. a vector space of form
\[
(\mathbb{R}^2)^k = \bigoplus_k (\mathbb{R}^2) := \mathbb{R}^2 \oplus \cdots \oplus \mathbb{R}^2.
\]

After journalized transactions, accountants post the journal entries to special ledger or general ledger or subsidiary ledger. This process extract the actions in some specific groups of action concerning some debit or credit amounts.

**Corollary 5.4** A subsidiary ledger or general ledger can be presented as a vector subspace of the vector space, generated by the corresponding T-accounts, i.e. a subspace of \((\mathbb{R}^2)^k\).

5.3 Basic Accounting Equation

In accounting, the basic accounting like: Assets(A), Liabilities(L), Owner’s Equity(OE): Capital(C), Revenue(R), Expenses(E) and Drawing(D) are related in the so called the basic accounting equation
\[ A = L + OE, \]
or the extended basic accounting equation
\[ A = L + R - E + C - D. \]
Proposition 5.1 The basic accounting equations and the extended accounting equations are systems of linear equations on the vector space of all accounts.

Proof Indeed each of the accounts is a 1-dimensional real vector space. And the equations are written for each debit or credit side.

5.4 Worksheet

In accounting (see [5]), a worksheet is a table of accounts with the pair of columns representing debit and credit sides of Trial Balance, Adjustments, Adjusted Trial Balance, Income Statements, and Balance Sheet.

Proposition 5.2 Each of the quantities: Trial Balance, Adjustments, Adjusted Trial Balance, Income Statements, and Balance Sheet can be represented by a vector subspace. A worksheet is a sum of such the vector subspaces.

Proof The table in the worksheet is exactly the sum of columns. Each of column represents a debit or credit of some account.

We fix a standard basis for each of the listed subspaces and denote the vector representing this worksheet as a column X.

5.5 Business Action

At the end of an accounting cycle, accountants transfer the net income to the capital for the next accounting cycle (see [5]). In our model, we have therefore the following statement.

Proposition 5.3 The process of transferring an amount from one place to another place in the worksheet is some linear operator in the worksheet vector space.

Indeed we have often exactly the same quantities at the initial places and at the targeting places.

Corollary 5.5 For a business action, there is a linear map and its corresponding matrix A, such that X is the input worksheet and AX is the output worksheet.

5.6 Leontief Models

We consider in the economics (see [5]), the models of Leontief: in one case - the closed model, when the output equal to the input, or in another words, the consumption equals to the production. And in another case - the open model, when a part of production is consumed by the producers and the another part of production is consumed by external bodies.

5.6.1 Closed Leontief Model $AX = X$

In this model, we have a worksheet $X$ as the input vector and a business operation, represented by a matrix $A$, the output vector is $AX$. The equation of the model is

$$AX = X.$$
The equation is written in equivalent form of a homogeneous system

\[(I - A)X = 0.\]

The following result is well-known in economics.

**Theorem 5.1** If the entries of the output-input matrix \(A\) are positive and if the sum of each column of \(A\) equal 1, then this system has one-parameter solution.

Remark that, the solution can be obtained by the well-known **Quantum Gauss-Jordan Elimination** procedure.

### 5.6.2 Open Leontief Model \(AX + D = X\)

In the open model, the vector \(D\) is called the *demand vector*, we can rewrite the system in form

\[(I - A)X = D.\]

It is also well-known the following result.

**Theorem 5.2** If the entries of the output-input matrix \(A\) are positive and if the sum of each column of \(A\) less than 1, then the matrix \(I - A\) is invertible and this system has a unique solution

\[X = (I - A)^{-1}D.\]

Remark once again that, the solution can be obtained by the well-known **Quantum Gauss-Jordan Elimination** procedure.

In actual accounting, there are some computer programs for solving the equations in excel, what we often use. If the size of matrix \(A\) is \(n \times n\) then in classical computation estimates of \(2^n\) computations. In this paper, we proposed a simulation of accounting by quantum computing, which let to solve system faster, estimated of order \(2^{n/2}\).

### 5.7 Complexification

In the previous model, debit and credit are deterministic quantities. In practice, there are some case, especially in finance and stock-markets, we use stochastic model. In that case, we extend the model of T-account from \(\mathbb{R}^2\) to complex plane \(\mathbb{C}^2\).

**Definition 5.2** A state of a T-account is a vector with complex components

\[\alpha|0\rangle + \beta|1\rangle \in \mathbb{C}^2,\]

where

\[|\alpha|^2 + |\beta|^2 = 1.\]

The number \(|\alpha|^2\) is the probability, that we have an amount in debit side, and the number \(|\beta|^2\) is the probability, that we have an amount in credit side.

**Corollary 5.6** An amount

\[x\alpha|0\rangle + y\beta|1\rangle\]
in a T-account is a distribution, that we have amount $x$ on debit side with probability $|\alpha|^2$ and have amount $y$ on credit side with probability $|\beta|^2$.

6 Simulation of Accounting on Quantum Computers

In accounting (see [5]), the normal balance of an account is on the side, where an increase in account is recorded. Trial balance of an account is the maximum part amounts, that balance debit and credit sides of that account. In general, trial balance is a list of accounts and their balances at a given time.

**Theorem 6.1** The decomposition of a T-account into trial balance and normal balance can be obtained by applying the 2-qubit CNOT gate to that decomposed account into the sum of two accounts.

**Proof** Suppose that, the T-account is of the form $|a\rangle|b\rangle$, i.e. has amount $a$ on the debit side and amount $b$ on the credit side. If $a \geq b$, we have a debit normal balance $a - b$. This means that

$$|a\rangle|b\rangle = (a - b)|0\rangle + b|1\rangle.$$ 

And if $a \leq b$, we have a credit normal balance $b - a$ and

$$|a\rangle|b\rangle = (b - a)|0\rangle + a|1\rangle.$$ 

These operation are well provided on quantum computers.

In business and accounting, we often need to move one amount from one account to another. For examples, at the end of an accounting cycle, all temporary accounts should be closed and move net income to the capital account of the next accounting cycle.

**Theorem 6.2** Transfering an amount from one account to another account can be simulated by the rule of tensor product of accounts.

**Proof** In quantum computers, the 2-dimensional vector spaces represent accounts, that are multiplied as tensor products. The coefficient of one factor is transferred to the factor of another account vector spaces.

$$\cdots \otimes |vk\rangle \otimes \cdots \otimes |vj\rangle \otimes \cdots = \cdots \otimes |v\rangle \otimes \cdots \otimes |kvj\rangle \otimes \cdots.$$ 

This simulates the process of transferring an amount from one account to another.

In accrual-basis accounting, transaction, that change a company’s financial statements are recorded in the period. In which, the events occurred, see [5]. There are four types of accrued accounts: prepaid expenses, unearned revenues, accrued revenues, accrued expenses. **Prepaid expenses** are the expenses paid in cash and recorded as assets, before they are used for consumed. **Unearned revenues** are revenues received in cash and recorded as liabilities and recorded, before revenues are earned. **Accrued expenses** are the expenses incurred, but not yet paid in cash or recorded. **Accrued revenues** are the revenues earned, but not yet received in cash or recorded. For these accounts, we need some adjustments by adding some adjustment accounts. **Adjustment entries** are made at the end of an account-
The adjustment process in accounting can be obtained by applying a 2-qubit CNOT gate.

Proof An adjustment consists of decomposing a normal balance into a sum of two subamounts. Then keep one component (for incurred expenses, . . .), and rotate the second component into opposite side (from debit to credit side). In mathematical language, we have

\[
\begin{align*}
|a\rangle|b\rangle &= |a_1\rangle|b\rangle + (a - a_1)|b\rangle, \\
|a\rangle|b\rangle &= |a\rangle|b_1\rangle + |a\rangle(b - b_1)\rangle.
\end{align*}
\]

or

Twist one account and keep another account is simulated be a 2-qubit C-NOT gate.

\[\text{Theorem 6.4} \quad \text{In Complexified accounting (i.e. stochastic accounting), the rotation of an account into the angle of } \frac{\pi}{4} \text{ is simulated by Hadamard gate and flip between amounts in the credit and debit sides.}\]

Proof The matrix of the Hadamard gate is decomposed into the product of the matrix of flip the basis vectors and the rotation matrix:

\[
H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.
\]

Acknowledgments The work was partially reported in the Mathematical Club and Seminar of Topology and Geometry, Institute of Mathematics in Hanoi. We thank the peoples, who attended for stimulating comments.

The paper was done in the framework of a project of research group “Noncommutative Geometry and Topology”, at VIASM. The first author thanks the Institute for a scientific stay support and Thang Long University for providing two invited lectures at Information Seminar.

References

1. Diep, D.N.: Quantum computers and related mathematical structures. J. Math. Appl. 2(1), 79-94 (2004) in Vietnamese
2. Press, W.H., Teukolsky, S.A., Velterling, W.T., Flannery, B.P.: Numerical Recipes in C, Cambridge University Press
3. Ekert, A., Hayden, P., Inamori, H.: Basis Concepts In Quantum Computation, Centre for Quantum Computation. University of Oxford, Oxford (2000)
4. Kaye, P., Laflamme, R., Mosca, M.: An Introduction to Quantum Computing, Oxford University Press
5. Weygandt, J., Kieso, D., Kimmel, P. Accounting Principles, 6th edn, p. 1138. Wiley, USA (2002)