04-93

Dehaene J., De Moor B., ``Stabilizer states, conditional Clifford operations, and their possible role in quantum computation", in *Proc. of MTNS 2004*, Leuven, Belgium, Jul. 2004, pp. 127_1-127_13., Lirias number: 180624.

Stabilizer states, conditional Clifford operations, and their possible role in quantum computation.

Jeroen Dehaene and Bart De Moor

Katholieke Universiteit Leuven, ESAT-SCD,
Kasteelpark Arenberg 10,
B-3001 Leuven, Belgium.

email: jeroen.dehaene@esat.kuleuven.ac.be

abstract: We discuss a possible road towards new quantum algorithms based on earlier work on stabilizer states and Clifford operations [1], using linear algebra in vector spaces over finite fields.

1 Introduction

We work with the following (fairly standard) setting for quantum computation. The computer consists of n qudits, i.e. quantum systems with a d-dimensional Hilbert space. We will mostly focus on qubits ($d = 2$) but most of the theory can be generalized for d prime and we are working on generalizations for general values of d. The initial state is a basis state reflecting the classical input (possibly including a fixed ancilla part). The program consists of one or two-qubit unitary operations applied to the state (or rather computes classically which operations to apply as a function of n and possibly the classical input), and the output is read by a final measurement of the qudits. The complexity of the algorithm is roughly speaking determined by the number of unitary operations required as a function of n. Only a few quantum algorithms exist with an exponential speed up over the best known classical algorithms. The best known example is Shor's algorithm for factoring integers [3].

The search for new quantum algorithms can be viewed as a trade-off between complexity and transparency. The degrees of freedom in finding sequences of unitary operations that lead to interesting results are vast but untransparent. For this reason it is necessary to restrict the freedom in a way that leads to mathematical transparency and enables to control or understand what the algorithm is doing, without losing the complexity that is necessary to prevent straightforward simulation on a classical computer. A simple (but not very general) example of transparent mathematics without allowing efficient simulation is the efficient quantum realization of a Fourier transformation, which is the heart of Shor's quantum algorithm. An at first sight appealing path to come up with more general settings is to work with Clifford operations acting on stabilizer states. Stabilizer states are joint eigenvectors with eigenvalue 1 of maximal commuting subgroups of the n-qubit Pauligroup. (Generalizations to qudits are possible). Clifford operations are unitary operations that map the Pauligroup to itself under conjugation. Clifford operations map stabilizer states to stabilizer states. Clifford operations and stabilizer states can be elegantly desribed by linear (and quadratic) algebra over $\mathbb{Z}_2$ [1]. However, working

with only stabilizer states and Clifford operations is too restrictive to disallow efficient simulation on a classical computer (see for instance [2]).

In this paper we add conditional Clifford operations to the above picture. Conditional operations are conditional in the same sense as the well known conditional not operation (CNOT). The operation leaves half of the basis states invariant (say, if the first qubit is $|0\rangle$) and applies an $(n-1)$-qubit Clifford operation to the other basis states. Multiple conditions are possible but below we will restrict ourselves to single conditions. In matrix terms a conditional Clifford operation with the condition that the first qubit is 1, can be thought of as a block matrix

$$\begin{bmatrix} I & 0 \\ 0 & Q_c \end{bmatrix}$$

where the partition corresponds to the value of the first qubit and Q_c is a Clifford operation. (Note that the overal phase of Q_c does matter).

The effect of a conditional Clifford operation remains mathematically transparent if the part of the state on which it acts (the component along the basis states satisfying the condition) is a stabilizer state. If the full state is a stabilizer state, this is the case. However the new full state need not be a stabilizer state again. (There is no compelling reason to stick to stabilizer states but the path looks fruitful). To study when a stabilizer state is mapped to a stabilizer state, the mathematical setting of [1] in which the coefficients of a stabilizer state (in the standard basis expansion) are described by quadratic forms over $\mathbb{Z}_2^n$, proves very useful. Different schemes can be worked out where a subset of the stabilizer states ($\mathcal{S}$) is preserved by a subset of the Clifford group supplemented with a number of conditional Clifford operations which together generate a group ($\mathcal{G}$).

However in this setting the sequence of stabilizer states obtained by the consecutive Clifford and conditional Clifford operations can still be efficiently simulated. To avoid this pitfall we propose a scheme in which simulation on an exponentially large (with growing n) subset of $\mathcal{S}$ would be necessary to simulate the algorithm. This is achieved as follows. The total action of the program (without the final measurement) on the initial state can be described by a unitary matrix. The columns of this matrix are the final states for the different possible (standard basis) input states. If the program consists of a first operation G_0 mapping the initial states into $\mathcal{S}$, followed by operations in $\mathcal{G}$, this unitary matrix can be written as G_pG_0 where $G_p \in \mathcal{G}$. We now add an extra initial transformation H_0 yielding the total operation $G_pG_0H_0$. The idea is that G_0H_0 does not take the basis states into $\mathcal{S}$ such that straightforward simulation for a given basis state is no longer possible. However, we can think of H_0 as acting on the right on G_pG_0. The final states for the different possible inputs are then the columns of $G_pG_0H_0$ which are linear combininations of (an exponential number of) columns of G_pG_0. The idea is that understanding the action of G_p should allow one to understand the complete program $G_pG_0H_0$ without allowing straightforward simulation.

In sections 3 and 4 we give an example of such a setting, which has not led to new efficient algorithms, but which we hope provides inspiration for further progress. First, in section 2

we recall some of the material of [1], to describe Clifford operations and stabilizer states with binary linear and quadratic algebra.

2 Stabilizer states, Clifford operations and linear and quadratic algebra over $\mathbb{Z}_2$

In this section we state some definitions and briefly review results of [1].

The n-qubit Pauli group $\mathcal{P}_n$ consists of tensor products of n matrices from the set

$$\tau_{00} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \tau_{01} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \tau_{10} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \tau_{11} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}.$$

and an overal factor $1,i,-1$ or $-i$. Each element is uniquely coded by specifying a $2n$-dimensional binary vector $[u^T v^T]^T$ with $u, v \in \mathbb{Z}_2^n$ and two extra bits δ and ϵ, together coding the $2^n \times 2^n$ complex matrix

$$i^\delta (-1)^\epsilon \tau_{u_1 v_1} \otimes \cdots \otimes \tau_{u_n v_n}.$$

A Pauli group element is Hermitian iff $\delta = u^T v$.

Below, a stabilizer group is a commutative subgroup of the Pauli group with 2^n Hermitian elements and not containing $iI, -I$ or $-iI$. Such a group is generated by n independent elements, and can be represented by a $2n \times n$ matrix S and an n-dimensional vector b, coding the values of u,v and ϵ for the n generators. Such a stabilizer group uniquely codes a stabilizer state, i.e. an eigenvector with eigenvalue 1 of all group elements. We will denote such a state as $|\psi_{S,b}\rangle$.

The Clifford group consists of unitary $2^n \times 2^n$ operations (where an overall phase does not matter), that map the Pauli group onto itself under conjugation: $Q\mathcal{P}_n Q^\dagger = \mathcal{P}_n$.

A Clifford group element (or Clifford operation) Q can be represented by a $2n \times 2n$ symplectic matrix C and a $2n$-dimensional vector h, coding the images of a standard set of generators of the Pauli group. We will denote this Clifford operation as $Q_{C,h}$.

The product of two Clifford operations and the action of a Clifford operation can be described entirely in terms of linear and quadratic operations in vector spaces over $\mathbb{Z}_2$.

To this end, we first introduce the following notation, simplifying the formulas.

$$
\begin{aligned}
U &= \begin{bmatrix} 0 & I_n \\ 0 & 0 \end{bmatrix} \\
\bar{U} &= \begin{bmatrix} U & 0 \\ 0 & 1 \end{bmatrix} \\
\bar{C} &= \begin{bmatrix} C & 0 \\ d^T & 1 \end{bmatrix} \text{ where} \\
d &= \mathcal{V}_{\text{diag}}(C^T U C) \\
\bar{h} &= \begin{bmatrix} h \\ 0 \end{bmatrix} \\
\bar{S} &= \begin{bmatrix} S \\ f^T \end{bmatrix} \text{ where} \\
f &= \mathcal{V}_{\text{diag}}(S^T U S)
\end{aligned}
$$

where $\mathcal{V}_{\text{diag}}(X)$ is a vector with the diagonal elements of X.

Now if $Q_{C_{21},h_{21}} = Q_{C_2,h_2} Q_{C_1,h_1}$ then

$$
\begin{aligned}
\bar{C}_{21} &= \bar{C}_2 \bar{C}_1 \\
\bar{h}_{21} &= \bar{h}_1 + \bar{C}_1^T \bar{h}_2 + \mathcal{V}_{\text{diag}}(\bar{C}_1^T \mathcal{P}_{\text{lows}}(\bar{C}_2^T \bar{U} \bar{C}_2) \bar{C}_1)
\end{aligned}
$$

where $\mathcal{P}_{\text{lows}}(X)$ is the strictly lower triangular part of X. (Note that such an operation appears naturally in the descriptions of quadratic forms, which cannot be described by $q(x) = x^T A x$ with A symmetric when working in binary vector spaces.)

Furthermore, $Q_{C,h}|\psi_{S,b}\rangle = |\psi_{S',b'}\rangle$ with

$$
\begin{aligned}
\bar{S}' &= \bar{C} \bar{S} \\
b' &= b + S^T h + \mathcal{V}_{\text{diag}}(\bar{S}^T \mathcal{P}_{\text{lows}}(\bar{C}^T \bar{U} \bar{C}) \bar{S})
\end{aligned}
$$

The representation of a stabilizer by a set of generators is not unique. We have $|\psi_{S',b'}\rangle = |\psi_{S,b}\rangle$ if there exists an invertible R such that

$$
\begin{aligned}
\bar{S}' &= \bar{S} R \\
b' &= R^T b + \mathcal{V}_{\text{diag}}(R^T \mathcal{P}_{\text{lows}}(\bar{S}^T \bar{U} \bar{S}) R)
\end{aligned}
$$

The coefficients of a stabilizer state in a standard basis expansion can be described by quadratic forms over $\mathbb{Z}_2^n$. Here we give only the result for so called graph states described by

$$
S = \begin{bmatrix} Z \\ I \end{bmatrix}
$$

where Z is a symmetric matrix with zero diagonal. Such a state has the standard basis expansion

$$|\psi_{S,b}\rangle = 2^{-n/2} \sum_{x \in \mathbb{Z}_2^n} (-1)^{x^T \mathcal{P}_{\text{lows}}(Z) x + b^T x} |x\rangle$$

Throughout this text we will mostly drop the factor $2^{-n/2}$.

3 A concrete set of stabilizer states and Clifford and conditional Clifford operations

As a concrete example of the general ideas explained in section 1 we consider the subset $\mathcal{S}$ of stabilizer states $|\psi_{Z,b_1,b_2,c}\rangle = (-1)^c |\psi_{S,b}\rangle$ with

$$\begin{aligned} S &= \begin{bmatrix} Z & 0 \\ 0 & 0 \\ I & 0 \\ 0 & I \end{bmatrix} \\ b^T &= [b_1^T b_2^T] \end{aligned}$$

where Z is a zero diagonal symmetric (binary) $m \times m$ matrix and the subscripts 1 and 2 refer to the first m and last $n - m$ qubits respectively. Note that $|\psi_{Z,b_1,b_2,c}\rangle$ is a graph state. As this will be necessary below we also introduce an overall phase $(-1)^c$ of the state. The states considered then have the following standard basis expansion

$$|\psi_{Z,b_1,b_2,c}\rangle = \sum_{x \in \mathbb{Z}_2^n} (-1)^{x_1^T \mathcal{P}_{\text{lows}}(Z) x_1 + b_1^T x_1 + b_2^T x_2 + c} \left| \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right\rangle$$

The following Clifford operations map $\mathcal{S}$ onto itself.

(1) The operations $\mathcal{L}_L = Q_{C,h}$ with

$$\begin{aligned} C &= \begin{bmatrix} I & L^T & 0 & 0 \\ 0 & I & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & L & I \end{bmatrix} \\ h &= 0 \end{aligned}$$

These operations map a state $|\psi_{Z,b_1,b_2,c}\rangle$ to $|\psi_{Z',b_1',b_2',c'}\rangle$ with

$$\begin{aligned} Z' &= Z \\ b_1' &= b_1 + L^T b_2 \\ b_2' &= b_2 \\ c' &= c \end{aligned}$$

(2) The operations $\mathcal{F}_F = Q_{C,h}$ with

$$\begin{aligned} C &= \begin{bmatrix} I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \\ 0 & F^T & I & 0 \\ F & 0 & 0 & I \end{bmatrix} \\ h &= 0 \end{aligned}$$

These operations map a state $|\psi_{Z,b_1,b_2,c}\rangle$ to $|\psi_{Z',b_1',b_2',c'}\rangle$ with

$$\begin{aligned} Z' &= Z \\ b_1' &= b_1 + ZF^T b_2 \\ b_2' &= b_2 \\ c' &= c + b_2^T F b_1 + b_2^T F \mathcal{P}_{\text{lows}}(Z) F^T b_2 \end{aligned}$$

(3) The operations $\mathcal{G}_G = Q_{C,h}$ with

$$\begin{aligned} C &= \begin{bmatrix} I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & G & 0 & I \end{bmatrix} \\ h &= 0 \end{aligned}$$

These operations map a state $|\psi_{Z,b_1,b_2,c}\rangle$ to $|\psi_{Z',b_1',b_2',c'}\rangle$ with

$$\begin{aligned} Z' &= Z \\ b_1' &= b_1 \\ b_2' &= b_2 \\ c' &= c + b_2^T \mathcal{P}_{\text{lows}}(G) b_2 \end{aligned}$$

(4) The operations $\mathcal{J}_J = Q_{C,h}$ with

$$\begin{aligned} C = \quad &= \begin{bmatrix} I & 0 & J & 0 \\ 0 & I & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \end{bmatrix} \\ h \quad &= 0 \end{aligned}$$

These operations map a state $|\psi_{Z,b,c}\rangle$ to $|\psi_{Z',b_1',b_2',c'}\rangle$ with

$$\begin{aligned} Z' &= Z + J \\ b_1' &= b_1 \\ b_2' &= b_2 \\ c' &= c \end{aligned}$$

(5) The operations $\mathcal{K}_{h_x,h_z} = Q_{C,h}$ with

$$\begin{aligned} C &= \begin{bmatrix} I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \end{bmatrix} \\ h^T &= [h_{x1}^T h_{x2}^T h_{z1}^T h_{z2}^T] \end{aligned}$$

These operations map a state $|\psi_{Z,b_1,b_2,c}\rangle$ to $|\psi_{Z',b_1',b_2',c'}\rangle$ with

$$\begin{aligned} Z' &= Z \\ b_1' &= b_1 + Zg_1 + h_{z1} \\ b_2' &= b_2 + h_{z2} \\ c' &= c + h_{x1}^T \mathcal{P}_{\text{lows}}(Z) h_{x1} + h_{x1}^T b_1 + h_{x2}^T b_2 \end{aligned}$$

As described above, we extend this set of operations with conditional Clifford operations. We consider a conditional Clifford operation $\mathcal{C}_{Q_c}^{(k)}$ applying the $(n-1)$-qubit Clifford operation Q_c if the k-th qubit is 1. As above this means that Q_c is applied to the component ψ in the subspace spanned by the basis states for which qubit k is 1 ($k \leq m$). As the overall phase of Q_c now matters we will assume that the top left element of Q_c is real and positive (in the cases below this element is never 0). Assume without loss of generality that $k = 1$, then the state to which Q_c is applied is

$$|\psi_{\tilde{Z},\tilde{b}_1+\tilde{z},b_2,c+\beta}\rangle = \sum_{\tilde{x}_1 \in \mathbb{Z}_2^{m-1}} \sum_{x_2 \in \mathbb{Z}_2^{n-m}} (-1)^{\tilde{x}_1^T \mathcal{P}_{\text{lows}}(Z)\tilde{x}_1 + (\tilde{b}_1+\tilde{z})^T \tilde{x}_1 + b_2^T x_2 + c + \beta} \left| \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right\rangle$$

where

$$\begin{aligned} Z &= \begin{bmatrix} 0 & \tilde{z}^T \\ \tilde{z} & \tilde{Z} \end{bmatrix} \\ b_1^T &= [\beta \;\; \tilde{b}_1] \end{aligned}$$

When Q_c is applied to this state a new stabilizer state will be obtained. However, as mentioned above, we want $\mathcal{C}_{Q_c}^{(k)}$ to transform the complete n-qubit state into a new stabilizer state. This is achieved if Q_c leaves the quadratic part $\tilde{Z}$ unchanged.

If Q_c transforms the $(n-1)$-qubit state $|\psi_{\tilde{Z},\tilde{b}_1+\tilde{z},b_2,c+\beta}\rangle$ into $|\psi_{\tilde{Z},\tilde{b}_1+\tilde{z}+\bar{\tilde{b}}_1,b_2,c+\beta+\bar{\beta}}\rangle$, (with the same quadratic part), the conditional Clifford operation $\mathcal{C}_{Q_c}^{(k)}$, transforms $|\psi_{Z,b_1,b_2,c}\rangle$ into $|\psi_{Z',b_1',b_2',c'}\rangle$ with

$$\begin{aligned} \tilde{z}' &= \tilde{z} + \bar{\tilde{b}}_1 \\ \tilde{Z}' &= \tilde{Z} \\ \beta' &= \beta + \bar{\beta} \\ \tilde{b}_1' &= \tilde{b}_1 \\ b_2' &= b_2 \end{aligned}$$

Now, we apply this idea to derive conditional operations from the operations $\mathcal{L}$, $\mathcal{F}$ and $\mathcal{G}$ defined above. (The operations $\mathcal{K}$ and $\mathcal{J}$ are not considered since conditional $\mathcal{K}$ operations yield Clifford operations and $\mathcal{J}$ does not leave the quadratic part invariant). First, the operations $\mathcal{L}$ and $\mathcal{F}$ can be made into $(n-1)$-qubit operations by choosing $\tilde{L}$ and $\tilde{F}$ to be $(n-m)\times(m-1)$ instead of $(n-m)\times m$. Then applying the above ideas one obtains the following conditional Clifford operations

(6) The operations $\mathcal{L}_{\tilde{L}}^{(k)}$ result in

$$\begin{aligned}\tilde{z}' &= \tilde{z}^T + b_2^T\tilde{L}\\ \beta' &= \beta\end{aligned}$$

(7) The operations $\mathcal{F}_{\tilde{F}}^{(k)}$ result in

$$\begin{aligned}\tilde{z}' &= \tilde{z}^T + b_2^T\tilde{F}\tilde{Z}\\ \beta' &= \beta + b_2^T\tilde{F}(\tilde{b}_1+\tilde{z}) + b_2^T\tilde{F}\mathcal{P}_{\text{lows}}(\tilde{Z})\tilde{F}^T b_2\end{aligned}$$

(8) The operations $\mathcal{G}_G^{(k)}$ result in

$$\begin{aligned}\tilde{z}' &= \tilde{z}^T\\ \beta' &= \beta + b_2^T\mathcal{P}_{\text{lows}}(G)b_2\end{aligned}$$

The superscipt (k) refers to the qubit on which the condition is imposed. Note that the above descriptions are valid for general values of k if $\tilde{z}$ is defined to be the k-th column of Z without the k-th element, $\tilde{Z}$ is Z without the k-th row and column and β is the k-th element of b_1.

By composing the above 5 Clifford operations (with $h_{z2}=0$ in the operations $\mathcal{K}_{h_x,h_z}$, see below), and 3 conditional Clifford operations one can code the transformation $\mathcal{C}_{R,C,f,g,\phi}$ which tranforms $|\psi_{Z,b_1,b_2,c}\rangle$ into $|\psi_{Z',b_1',b_2',c'}\rangle$ with

$$\begin{aligned}Z' &= R(b_2)^T Z R(b_2) + C(b_2)\\ b_1' &= R(b_2)^T b_1 + R(b_2)^T Z f(b_2) + \mathcal{V}_{\text{diag}}(R(b_2)^T\mathcal{P}_{\text{lows}}(Z)R(b_2)) + g(b_2)\\ b_2' &= b_2\\ c' &= c + f(b_2)^T b_1 + f(b_2)^T\mathcal{P}_{\text{lows}}(Z)f(b_2) + \phi(b_2)\end{aligned} \tag{1}$$

where R is an arbitrary maps from $\mathbb{Z}_2^{n-m}$ into the invertible $m\times m$ matrices, C is an arbitrary map from $\mathbb{Z}_2^{n-m}$ into the zero diagonal symmetric $m\times m$ matrices, f and g are arbitray maps from $\mathbb{Z}_2^{n-k}$ into $\mathbb{Z}_2^k$ and ϕ is an arbitrary map from $\mathbb{Z}_2^{n-m}$ into $\mathbb{Z}_2$.

To prove this fact, we first show the effect of composing $\mathcal{C}_{R,C,f,g,\phi}$ with the 8 operations defined above, and then show that it can be decomposed into a product of such operations.

(1) Composing $\mathcal{L}_L$ with $\mathcal{C}_{R,C,f,g,\phi}$ yields $\mathcal{C}_{R',C',f',g',\phi'}$ with

$$\begin{aligned} R'(b_2) &= R(b_2) \\ C'(b_2) &= C(b_2) \\ f'(b_2) &= f(b_2) \\ g'(b_2) &= g(b_2) + L^T b_2 \\ \phi'(b_2) &= \phi(b_2) \end{aligned}$$

(2) Composing $\mathcal{F}_F$ with $\mathcal{C}_{R,C,f,g,\phi}$ yields $\mathcal{C}_{R',C',f',g',\phi'}$ with

$$\begin{aligned} R'(b_2) &= R(b_2) \\ C'(b_2) &= C(b_2) \\ f'(b_2) &= f(b_2) + R(b_2)F^T b_2 \\ g'(b_2) &= g(b_2) + C(b_2)F^T b_2 \\ \phi'(b_2) &= \phi(b_2) + b_2^T F \mathcal{P}_{\text{lows}}(C(b_2))Fb_2 + b_2^T F g(b_2) \end{aligned}$$

(3) Composing $\mathcal{G}_G$ with $\mathcal{C}_{R,C,f,g,\phi}$ yields $\mathcal{C}_{R',C',f',g',\phi'}$ with

$$\begin{aligned} R'(b_2) &= R(b_2) \\ C'(b_2) &= C(b_2) \\ f'(b_2) &= f(b_2) \\ g'(b_2) &= g(b_2) \\ \phi'(b_2) &= \phi(b_2) + b_2^T \mathcal{P}_{\text{lows}}(G)b_2 \end{aligned}$$

(4) Composing $\mathcal{J}_J$ with $\mathcal{C}_{R,C,f,g,\phi}$ yields $\mathcal{C}_{R',C',f',g',\phi'}$ with

$$\begin{aligned} R'(b_2) &= R(b_2) \\ C'(b_2) &= C(b_2) + J \\ f'(b_2) &= f(b_2) \\ g'(b_2) &= g(b_2) \\ \phi'(b_2) &= \phi(b_2) \end{aligned}$$

(5) Composing $\mathcal{K}_{h_x,h_z}$ (with $h_{z2} = 0$) with $\mathcal{C}_{R,C,f,g,\phi}$ yields $\mathcal{C}_{R',C',f',g',\phi'}$ with

$$\begin{aligned} R'(b_2) &= R(b_2) \\ C'(b_2) &= C(b_2) \\ f'(b_2) &= f(b_2) \\ g'(b_2) &= g(b_2) \\ \phi'(b_2) &= \phi(b_2) + h_{x1}^T \mathcal{P}_{\text{lows}}(C(b_2))h_{x1} + h_{x1}^T g(b_2) + h_{x2}^T b_2 \end{aligned}$$

Note that allowing $h_{z2} \neq 0$ would entail a change of b_2. However, while this would complicate the notation, this does not make a significant difference. Therefore we do not consider this case.

(6) Composing $\mathcal{L}_{\tilde{L}}^{(k)}$ with $\mathcal{C}_{R,C,f,g,\phi}$ yields $\mathcal{C}_{R',C',f',g',\phi'}$ with

$$\begin{aligned} R'(b_2) &= R(b_2) \\ C'(b_2) &= C(b_2) + e_k c(b_2)^T + c(b_2) e_k^T \\ f'(b_2) &= f(b_2) \\ g'(b_2) &= g(b_2) \\ \phi'(b_2) &= \phi(b_2) \end{aligned}$$

where e_k is the k-th column of the identity matrix and $c(b_2)$ is equal to 0 in its k-th entry and equal to $\tilde{L}^T b_2$ in the other entries.

(7) Composing $\mathcal{F}_{\tilde{F}}^{(k)}$ with $\mathcal{C}_{R,C,f,g,\phi}$ yields $\mathcal{C}_{R',C',f',g',\phi'}$ with

$$\begin{aligned} R'(b_2) &= R(b_2) r(b_2) \\ C'(b_2) &= r(b_2)^T C(b_2) r(b_2) \\ f'(b_2) &= f(b_2) \\ g'(b_2) &= g(b_2) + e_k (b_2^T \tilde{F} (\tilde{g}(b_2) + \tilde{c}) + b_2^T \tilde{F} \mathcal{P}_{\text{lows}}(\tilde{C}) \tilde{F}^T b_2 \\ \phi'(b_2) &= \phi(b_2) \end{aligned}$$

where $\tilde{g}$,$\tilde{C}$ and $\tilde{c}$ are defined in a similar way as $\tilde{b}_1$,$\tilde{Z}$ and $\tilde{z}$.

(8) Composing $\mathcal{G}_G^{(k)}$ with $\mathcal{C}_{R,C,f,g,\phi}$ yields $\mathcal{C}_{R',C',f',g',\phi'}$ with

$$\begin{aligned} R'(b_2) &= R(b_2) \\ C'(b_2) &= C(b_2) \\ f'(b_2) &= f(b_2) \\ g'(b_2) &= g(b_2) + e_k b_2^T \mathcal{P}_{\text{lows}}(G) b_2 \\ \phi'(b_2) &= \phi(b_2) \end{aligned}$$

Now we have to prove that composing the above 8 operations leads to Eq. (1). We only sketch the proof. If $m = n - 1$ (b_1 is one bit only), one can easily compose the basic operations to yield Eq. (1). This result can then first be extended for general m and n and affine dependence of R, C, f, g and ϕ on b_2. For general nonlinear dependence, the proof gets more involved, and the decomposition of Eq. (1) is no longer efficient. The idea is to first code an operation with say $C(b_2)$ zero for all values of b_2 except one. This can be done by composing operations which depend linearly on b_2 and making their effect cancel for all but one value of b_2.

4 An approach to quantum algorithms

As explained in the introduction, we consider a program $G_p G_0 H_0$. The operation G_0 takes a standard basis state $|y\rangle$ into the set $\mathcal{S}$. We will set G_0 equal to a Hadamard operation

on all qubits. The operation $G_p \in \mathcal{G}$ is a composition of Cliffford and conditional Clifford operations as defined above, $G_p = \mathcal{C}_{R,C,f,g,\phi}$. It takes states in $\mathcal{S}$ into $\mathcal{S}$ again. The program G_pG_0 can be efficiently simulated for a given input on a classical computer as all steps can be calculated on the binary representations. However, the operation H_0 is an extra operation which will be carried out first in the actual program. As a result G_0H_0 no longer takes the initial states into $\mathcal{S}$ avoiding the possibility of straightforward efficient simulation. But we can think of the final state for an input $|u\rangle$, i.e. column u of $G_pG_0H_0$, as a linear combination of the columns of G_pG_0 (these are the final states of the program without H_0), with the coefficients in column u of H_0. This will enable us to understand what the program does without being able to simulate it efficiently.

We first consider the program G_pG_0 acting on a basis state $|y\rangle$ (think of y as the column index of G_pG_0. The Hadamard transformation G_0 takes the state $|y\rangle$ with $y \in \mathbb{Z}_2$ to the state

$$|\psi_{0,y_1,y_2,0}\rangle = \sum_{x\in\mathbb{Z}_2^n} (-1)^{y^T x}|x\rangle$$

According to Eq. (1), the transformation $G_p = \mathcal{C}_{R,C,f,g,\phi}$ transforms this state to $|\psi_{Z(y),b_1(y),b_2(y),c(y)}\rangle$ with

$$\begin{aligned} Z(y) &= C(y_2) \\ b_1(y) &= R(y_2)^T y_1 + g(y_2) \\ b_2(y) &= y_2 \\ c(y) &= f(y_2)^T y_1 \end{aligned}$$

with standard basis expansion

$$\begin{aligned} &|\psi_{Z(y),b_1(y),b_2(y),c(y)}\rangle = \\ &\textstyle\sum_{v_1\in\mathbb{Z}_2^m} \sum_{v_2\in\mathbb{Z}_2^{n-m}} (-1)^{v_1^T \mathcal{P}_{\text{lows}}(C(y_2))v_1 + v_1^T R(y_2)^T y_1 + v_1^T g(y_2) + v_2^T y_2 + f(y_2)^T y_1 + \phi(y_2)} |v\rangle \end{aligned} \qquad (2)$$

Now, for the additional operation H_0 we take a Hadamard transformation on only the first m qubits. (In practice this means that the initial Hadamard transformation on all qubits is replaced by a Hadamard on the last $n-m$ qubits, but this is not the way to understand what the program does). With this operation H_0 included, column u (the final state for the actual input $|u\rangle$) is a linear combination with coefficients $(-1)^{u_1{}^T y_1}$ of the columns $[y_1^T u_2^T]^T$ of the original program G_pG_0. This yields a state

$$\sum_{v\in\mathbb{Z}_2^n} (-1)^{\gamma(u_2,v_1,v_2)} \sum_{y_1\in\mathbb{Z}_2^m} (-1)^{u_1^T y_1 + v_1^T R(u_2)^T y_1 + f(u_2)^T y_1} |v\rangle$$

where γ is some function of u_2,v_1 and v_2.

The sum (for given v_1,v_2) is nonzero if and only if

$$R(u_2)v_1 + f(u_2) + u_1 = 0 \tag{3}$$

This condition is independent of v_2. This means that measuring the first m qubits, deterministicly yields the answer $v_1 = R(u_2)^{-1}(f(u_2) + u_1)$.

If we think of u_2 as the input and u_1 as an ancilla which may be 0, and if $R(u_2) = I$, the program calculates the arbitrary boolean function f. However in this way the program can be efficiently simulated as we can keep track of f through the different steps.

We have not yet found a way of exploiting this scheme to come up with efficient new quantum algorithms. However, we still hope that our approach can provide inspiration to achieve this goal, mainly for the following reason. Imagine there were also a linear dependence on y_1 of the term C in Eq. (2). Such a dependence would lead to a quadratic systems of equations instead of the linear one in Eq. (3). If we now think of u_1 as the input and of u_2 together with the program as coding the quadratic system of equations to be solved, we have a quantum algorithm for solving a (yet unknown) class of systems of quadratic equations over $\mathbb{Z}_2^m$, without the a priori possiblility of straightforward efficient simulation. Actually we *can* make C in Eq. (2) depend linearly on y_1 (with a slight extension of the above operations) but up to now only in ways that lead to quadratic systems that are efficiently solvable classically.

5 Conclusion

We have presented an approach to finding new quantum algorithms. The approach is based on stabilizer states and Clifford operations with the addition of conditional Clifford operations. Straightforward efficient simulation on a classical computer is circumvented through the idea of an additional initial operation with the effect of combining the final states for an exponential number of original inputs. The approach has not led to new algorithms that perform better than their classical counterparts yet. But we have argued why we consider the approach as promising.

Acknowledgements

This research was supported by (1) The Research Council KUL: GOA-Mefisto666, GOA AMBioRICS (2) The Flemisch government: FWO: projects, G.0240.99 (multilinear algebra), G.0407.02 (support vector machines), G.0197.02 (power islands), G.0141.03 (Identification and cryptography), G.0491.03 (control for intensive care glycemia), G.0120.03 (QIT), G.0452.04 (new quantum algorithms), G.0499.04 (Robust SVM), research communities (ICCoS, ANMMM, MLDM); AWI: Bil. Int. Collaboration Hungary/ Poland; IWT: PhD Grants,GBOU (McKnow) (3) The Belgian Federal Science Policy Office: IUAP

P5/22 ('Dynamical Systems and Control: Computation, Identification and Modelling', 2002-2006) ; PODO-II (CP/40: TMS and Sustainability); (4) The European Union: FP5-Quprodis; ERNSI; Eureka 2063-IMPACT; Eureka 2419-FliTE;

References

[1] J. Dehaene and B. De Moor, "Clifford group, stabilizer states, and linear and quadratic operations over gf(2)," *Physical Review A*, vol. 68, no. 042318, 2003.

[2] D. Gottesman, "The heisenerg representation of quantum computers," in *Group22: Proceedings of the XXII International Colloquium on Group Theoretical Methods in Physics* (S.P. Corney, R. Delbourgo, and P.D. Jarvis, eds.), pp. 32–43, International Press, Cambridge MA, 1998. quant-ph/9807006.

[3] P.W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Journal of computing*, vol. 26, p. 1484, 1997.