Efficient Simulation of Stabilizer States Using the Graph State Approach

Ekaterina Taralova, University of Arizona, taralove@u.arizona.edu
Submitted parts {1.1, 1.3, 2, 3, 4}
David Butler, UCLA, dwbutler@ucla.edu
Submitted parts {1.2}
Albert Chang, University of California, Berkeley, albert568@yahoo.com
Submitted parts {none}
Developed during the 2004 Computing Beyond Silicon Summer School, Caltech, CA

Abstract: Currently, the way we represent stabilizer states is by hand. However, this method is neither efficient nor productive. Several algorithms have been proposed to address the issue of efficient simulation of stabilizer states, but no one has implemented them yet. This paper discusses how the graph state approach method is used to develop a software application that will make quantum state visualization fast and convenient.

1. Background theory

1.1 Quantum gates and stabilizers

The set of gates that each stabilizer group comprises of is shown in Table 1.

Identity	$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$	$I\|a\rangle = \|a\rangle$
Bit Flip	$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$	$X\|a\rangle = \|a \oplus 1\rangle$
Phase Flip	$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$	$Z\|a\rangle = (-1)^a \|a\rangle$
Bit & Phase	$Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} = iXZ$	$Y\|a\rangle = i(-1)^a \|a \oplus 1\rangle$

The Pauli matrices

Table 1

A stabilizer (S) of a quantum state |Φ> is a unitary transformation such that when applied to a state, it does not change it: S |Φ> = |Φ> . For example, the state $\frac{1}{\sqrt{2}}(|0_1 0_2\rangle + |1_1 1_2\rangle)$ is stabilized by the set of operators {II, X^1X^2, Z^1Z^2}, where X^1 operates on qubit 1, transforming the initial state to $\frac{1}{\sqrt{2}}(|1_1 0_2\rangle + |0_1 1_2\rangle)$, and X^2 operates on qubit 2, transforming it to $\frac{1}{\sqrt{2}}(|1_1 1_2\rangle + |0_1 0_2\rangle)$, which is the starting state. By definition, the operators II do not change the initial state.

1.2 Group Theory and its Relation to Quantum States

An n-qubit quantum state may be compactly represented using its generators. In group theory, the *generators* of a group is the subset of a group, the members of which can be used to generate all the other members of the group. Group theory is used in quantum mechanics to compactly represent quantum states. This is done by representing the quantum state as a group of *unitary operators*. (If quantum states are thought of as unit length vectors in complex Hilbert space, unitary operators may be thought of as matrices which rotate quantum states.) The group may be of arbitrary size, but is most compactly represented using the generating group. The members of such a group are termed generators. Most commonly, members of the *Pauli group* are used as the generators to describe a quantum state:

$$\{\pm[i]I, \pm[i]X, \pm[i]Y, \pm[i]Z\}$$

where the brackets [] indicate optional components. In principle, therefore, there are 16 possible generators in the Pauli group. Any composition of these could serve as valid basic input to the program. In practice, however, there are extra conditions imposed by the limited purpose of our program. For example, the multiplicative factors ($\pm[i]$) are ignored for the purposes of graphing the generators. Additionally, the operator -I is not allowed. Y can be generated using X and Z, so it is not used. Therefore, for our program, the only valid input is a composition of the following operators:

$$\{I, X, Z\}$$

1.3 Problem statement.

A quantum state comprised of 1 quantum bit (qubit) has two amplitudes: $\alpha| 1 > + \quad | 0 >$. Therefore, to describe a n qubit pure state we need $O(2^n)$ complex amplitudes. It looks impossible to describe the state concisely and even well-defined measures like entropy of entanglement are hard to compute. Proposed algorithms for qualifying entanglement using the Schmidt measure are considered computationally intractable. Overall, lack of efficiently computable entanglement measurements limits our understanding of the properties of entangled quantum states shared between more than two parties. However, Hein *et al.* proposed a study of the entanglement of stabilizer states using the graph state approach.

2. Approach

The graph state approach utilizes stabilizers states and makes it possible to generate efficient descriptions of a quantum state, since it requires only O(n) bits to specify a n qubit state. In addition, the states are easily computed, and the number of elementary operations scales polynomially with the log of the size of the Hilbert space.

Each state has a list of stabilizer elements, which can be reduced to a minimal number. For example, suppose we have the stabilizer list G = {II, XX, ZZ}. The minimum number of operations required to describe this list is {XX, ZZ}. This subset is called the generators of the stabilizer group. Every set of n generators describes a unique n-qubit quantum state, and in this example, this is the Bell state $\frac{1}{\sqrt{2}}(|0_1 0_2\rangle + |1_1 1_2\rangle)$.

For example, for the five qubit code, in order to perform single bit error correction, we have the following states:

$|0_L\rangle$ = 1/4 [|00000)
+ |11000) + |01100) + |00110) + |00011) + |10001)
- |10100) - |01010) - |00101) - |10010) - |01001)
- |01111) - |10111) - |11011) - |11101) - |11110)]

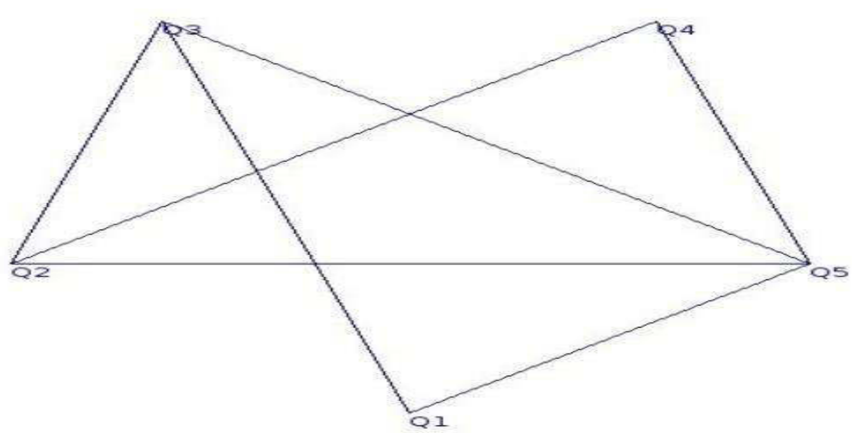
$|1_L\rangle$ = 1/4 [|11111)
+ |00111) + |10011) + |11001) + |11100) + |01110)
- |01011) - |10101) - |11010) - |01101) - |10110)
- |10000) - |01000) - |00100) - |00010) - |00001)]

This representation does not allow for the user to quickly determine any characteristics of the quantum state. However, using stabilizer states, we can represent this quantum state in terms of its generators elements:

g1 = XZZXI
g2 = IXZZX
g3 = XIXZZ
g4 = ZXIXZ
~Z = ZZZZZ

This description of the state still does not allow us to extract knowledge quickly and efficiently. Thus, we propose a software application that would be capable of efficiently simulating the stabilizer state using graphs, as shown on Figure 1.

Figure 1



3. Implementation

Computational part.

The computational part of the software application, named **qssv** (quantum stabilizer state visualization), is developed in the programming language C and it is open source, and it is published under the GNU license.

Currently, the application accepts a stabilizer generators list as its input. The input is checked for validity to ensure that the generators meet the requirements for stabilizing a state (see David Butler's paper for input checking). Afterwards, the N qubit input is inserted in a Nx2N matrix, called the "check matrix." After the check matrix has been populated, a variety of already developed (the algebra package from the Gandalf open source library) and of custom developed algebra routines are applied.

For the purpose of our application all linear algebra routines must perform binary arithmetic instead of addition and subtraction of decimal numbers. For this reason, all computational routines were developed and customized to perform the XOR operation.

Finally, the algorithm computes a Nx2N matrix with the following properties:

- the NxN part of the matrix is the identity matrix
- the other part of the matrix correlates the relationship between the stabilizer generators in the given state – the identity part corresponds to the operation X, while the second part of the matrix corresponds to the operator Z
- the correlation allows us to efficiently simulate the quantum state through a graph

The last property allows us to display a graph that is easy to interpret - there are only three key points to remember when analyzing the graph that simulates a stabilizer state:

- a vertex (qubit) denotes the X operator
- an edge denotes the Z operator
- no edge denotes the I operator

Graphical User Interface.

The information computed from the check matrix is then used by the graphical user interface module[1] to create a graph of the quantum state. This module is developed in Perl and GTK.

4. Future work

The following features are in their state of research:

- the application should take as an input a quantum state, be able to generate the stabilizer list, reduce it to the minimum number of generators required, and perform the computations – this will require a firm understanding of stabilizers, quantum gates and how to efficiently apply them to a quantum state.
- the application should be able to compute the degree of entanglement and display it in a manner that is easy to extract from the drawn graph state – this will require further research in quantum entanglement and ways of characterizing the strength of an entangled quantum state.

1 The GUI was developed by Tim Spriggs, Univesity of Arizona, tims@u.arizona.edu

- once the state is displayed, be able to apply local and global gates through the user interface, compute the result and display it
- all updates and the software application (including the source) will be published at http://cbsss2004.tajinc.org/

We would like to thank all the people who helped us during our research, especially Isaac Chuang (MIT), Dave Bacon (Caltech) and Panos Aliferis (Caltech).

References

D. Fattal, T. Cubitt, Y. Yamamoto, S. Bravyi, and I. Chuang, “Entanglement in the stabilier formalism,” June 23, 2004

M.Hein, J.Eisert, and H.J Briegel, Phys. Rev. A 69, 062311 (2004)

S. Aaronson, and D. Gottesman, “Improved Simulation of Stabilizer Circuits”

Nielsen, Michael and Isaac Chuang, “Quantum Computation and Quantum Information,” 2000, Cambridge University Press

J. Preskill, Physics 229 Lecture Notes, chapter 7

D. Gottesman, “Stabilizer Codes and Quantum Error Correction,” quant-ph/9705052 28 May 1997