A PROGRAM for GRAPH STATES

Albert H Chang
University of California, Berkeley
chang@berkeley.edu

I. GRAPH STATES?

What is a graph state? A graph state is another way to represent a vector space so that it will be easier for us to visualize the vector space. For our purposes, we use graph states to visualize our quantum states. But how?

First, we find a set of generators which stabilize the quantum state. Each generator will be a stabilizer of the quantum state with eigenvalue +1. For any *n*-qubit quantum state, we can find *n* linearly independent generators to describe the vector space.

After we find all the generators for that quantum state, then we will transform those generators into a matrix, which we call it CHECK MATRIX. The check matrix will have 2 parts, the X operator's part (on the left side, which indicate where X operators are located in the generators) and the Z operator's part (on the right side, which indicate where Z operators are located in the generators). If we see an X, then we will put a 1 on the left side according to its position in the generator; if we see a Z, then we will put a 1 on the right side according to its position in the generator; if we see an I, then we will put 0 on both sides; if we see a Y, then we will put 1 on both sides. Here is an example:

Let's say we have these 3 generators:

$g_1=X_1X_2X_3$, $g_2=Z_1Z_2I_3$, and $g_3=I_1Z_2Z_3$

then the check matrix will look like:

$$\left[\begin{array}{ccc:ccc} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{array}\right]$$

(one row is one generator)

After we make the check matrix, we have to change it to standard form (standard form: the left part of the matrix must be an identity matrix).

If there is any row which doesn't include X, then we need to apply Hadamard gate to switch column. For instance, the 2nd and the 3rd row don't have X in it, so we apply $H^{(2)}$ and $H^{(3)}$ to switch it. The resulting matrix will look like:

$$\left[\begin{array}{ccc:ccc} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \end{array}\right]$$

AFTER $H^{(2)}$ and $H^{(3)}$

After this, we can do the row operation:

ADD THE 2ND ROW TO THE THIRD ROW, we get:

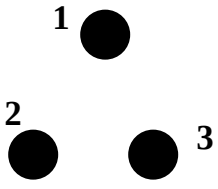
$$\left[\begin{array}{ccc:ccc} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \end{array}\right]$$

AFTER $H^{(2)}$ and $H^{(3)}$

Now, we have the check matrix in standard form. We can start graphing it. The 3 new generators after the transformation is

$g_1=X_1Z_2Z_3$, $g_2=Z_1X_2I_3$, and $g_3=Z_1I_2X_3$

This is a 3-qubit state, so first we draw 3 dots:



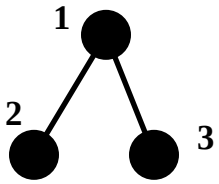
X means the qubit; and Z indicates if it is connected or not.

The first generator says $X_1Z_2Z_3$: X_1 means qubit one; Z_2Z_3 means qubit one is connected to qubit two and qubit three.

$Z_1X_2I_3$ means qubit two is connected to qubit one but not qubit three.

$Z_1I_2X_3$ means qubit three is also connected to qubit one but not qubit two.

So the final graph will be:



II. Valid INPUTs?

Are all inputs valid? We can't assume it is, because we are designing a program for someone who might have no idea how it works. As a result, before the inputs go into the actual program, we must check if the inputs are valid or our program may go into infinite loop or give us some incorrect results.

First, we have to check if the input generators are linearly independent, because we require a minimum group of generators which can generate all the other members in the vector space. If we have a linearly dependent generator in our group, then while we try to change our check matrix into standard form, it will produce a zero row which means we loose a generator in the process. As a result, we can't make a graph for it (because we loose some information about one of the qubits).

The way we use to check "linearly independent" in our program is doing our matrix transformation until we encounter a zero row then we will stop the process and return an error signal. If we do not encounter a zero row, then we will keep processing until we reach the standard form.

Second, we must check if all the generators commute with each other or not. Why do they need to commute with each other? The proof is as follows:

g_1 and g_2 are two generators (or stabilizers) for quantum state $|\varphi>$, and g_1 and g_2 anticommute with each other: $g_1g_2 = -g_2g_1$. We will say the quantum state $|\varphi>$ is not zero state or there is no point talking about it. Here is the contradiction:

$$g_1g_2 |\varphi>= g_1(g_2 |\varphi>) = g_1 |\varphi>=|\varphi>$$
$$= g_1 |\varphi>= g_2(g_1 |\varphi>) = -(g_1g_2 |\varphi>) = -|\varphi>$$

$$|\varphi>= -|\varphi>, |\varphi>= 0 \text{ ……. (X)}$$

So we need to have each generator commute with all other generators.

The way we check it is really easy. We need to have the check matrix G ready. Then we just do the following matrix multiplication:

$G\Lambda G^T = 0$, where Λ is a $2n \times 2n$ matrix $\Lambda = \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix}$.

With this, we can easily check if the generators commute with each other or not.

For example, if we have 3 generators:
$g_1=X_1X_2X_3$, $g_2=Z_1Z_2I_3$, and $g_3=I_1Z_2Z_3$

$$G = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}; \quad G^T = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix};$$

$$\Lambda = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

$G\Lambda G^T = 0$ (modulo two arithmetic: 1+1=0)

In real implementation, we use Gandalf for our linear algebra library so we can do any matrix operations easily.

III. Apply Unitary Gates to Graph State:

We have been able to use the stabilizer formalism to describe our quantum state. For an n-qubits quantum state, we will need n stabilizers to include the whole vector space. Then, we can graph the quantum state using the stabilizers. We will call the resulting graph the graph state of the original quantum state. Now, we are thinking about how to apply unitary quantum gates to our graph state, as we do to our quantum state. Unitary gate plays an important role in our quantum computations and future applications. For example, if we want to simulate a quantum circuit, we may need to put a unitary gate in our circuit to represent the unavoidable noise in real application.

At the first glance, we think we should first convert our stabilizers back into the quantum state representation. For example, if we have the following three stabilizers (or generators):

$g_1=X_1X_2X_3$, $g_2=Z_1Z_2I_3$, and $g_3=I_1Z_2Z_3$

and we want to apply Hadamard Gate to the second qubit. First, we convert it to our standard quantum state representation: $\frac{|000>+|111>}{\sqrt{2}}$; Then, apply Hadamard Gate to the second qubit. The resulting quantum state will be $\frac{|0+0>+|1-1>}{\sqrt{2}}$ with three generators: $g_1=X_1Z_2X_3$, $g_2=Z_1X_2I_3$, and $g_3=I_1X_2Z_3$. After we find out the all generators of the resulting quantum state then we can graph it using the rule we stated in the previous section.

It may be quite inefficient if we convert the generators back to standard representation, apply the unitary gates, find a new set of generators, and then graph it. Is there any way we can apply the unitary gates directly to every single generator instead of converting it back to the standard form? It might seem just a few more steps, but converting the generators back to the standard form is a really expensive computer procedure. We did not find any efficient methods to do the computation, so we must have a way to apply the unitary quantum gates directly to those stabilizers (or generators) as follows:

Let's say we want to apply unitary gate P to a quantum state $|\varphi>$ and let g be one of the generators of quantum state $|\varphi>$. As a result,

$$P|\varphi> = Pg|\varphi> = PgP^tP|\varphi>$$

We can tell that PgP^t becomes the new generator for new quantum state $P|\varphi>$, so we can do the same thing to every single generators of the quantum states, and we will get a new set of generators for the new quantum state after applying the unitary gate.

Let's go back to the quantum state with three generators $g_1=X_1X_2X_3$, $g_2=Z_1Z_2I_3$, and $g_3=I_1Z_2Z_3$ in the last example and use our new approach to solve the problem.

$$HXH^t = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\times\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}\times\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}=\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}=Z$$

$$HZH^t = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\times\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}\times\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}=\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}=X$$

So g_1 becomes $X_1Z_2X_3$, g_2 becomes $X_1Z_2X_3$ and g_3 becomes $I_1X_2Z_3$ (same as the 3 generators we predict before).

Let's say now we want to apply control-NOT on qubit one (control) and qubit two.

$$CX_{1}C^{t} = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{bmatrix}\times\begin{bmatrix}0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\\ 1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\end{bmatrix}\times\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{bmatrix}$$

$$=\begin{bmatrix}0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\\ 0 & 1 & 0 & 0\\ 1 & 0 & 0 & 0\end{bmatrix}=X_1X_2$$

$$CX_2C^t = \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}\times\begin{bmatrix}0&1&0&0\\1&0&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}\times\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}$$

$$= \begin{bmatrix}0&1&0&0\\1&0&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix} = X_2$$

$$CZ_1C^t = \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}\times\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&-1&0\\0&0&0&-1\end{bmatrix}\times\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}$$

$$= \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&-1&0\\0&0&0&-1\end{bmatrix} = Z_1$$

$$CZ_2C^t = \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}\times\begin{bmatrix}1&0&0&0\\0&-1&0&0\\0&0&1&0\\0&0&0&-1\end{bmatrix}\times\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}$$

$$= \begin{bmatrix}1&0&0&0\\0&-1&0&0\\0&0&-1&0\\0&0&0&1\end{bmatrix} = Z_1Z_2$$

The three generators will become

$g_1 = (X_1X_2)\, X_2\, X_3 = X_1\, I_2\, X_3$;
$g_2 = Z_1\, (Z_1Z_2)\, I_3 = I_1\, Z_2\, I_3$;
$g_3 = I_1\, (Z_1Z_2)\, Z_3 = Z_1\, Z_2\, Z_3$.

Here is the summery:

Operation	Input	Output
control-NOT	X_1	X_1X_2
	X_2	X_2
	Z_1	Z_1
	Z_2	Z_1Z_2
H	X	Z
	Z	X
S	X	Y
	Z	Z
X	X	X
	Z	$-Z$
Y	X	$-X$
	Z	$-Z$
Z	X	$-X$
	Z	Z

IV. Doing Measurement:

Till now, we let the closed quantum system evolve itself according to the unitary operations. It might be good if our quantum system does not interact with the rest of the universe because it will be easier for us to control. But many times, we want to find out what actually happens in our quantum system, so we may need to use some external tools to observe the system. After the observation, the system will no longer be closed and we will learn some details about the inside world. We call this doing measurement on the systems.

Let's review the three basic operators which we will use here to do measurement: X, Y, and Z operators.

X Operator: $\begin{bmatrix}0&1\\1&0\end{bmatrix}$;

Y Operator: $\begin{bmatrix}0&-i\\i&0\end{bmatrix}$;

Z Operator: $\begin{bmatrix}1&0\\0&-1\end{bmatrix}$

and

$x|+>=|+>,\ x|->=-|->$

$Y\frac{|0>+i|1>}{\sqrt{2}} = \frac{|0>+i|1>}{\sqrt{2}},\ Y\frac{|0>-i|1>}{\sqrt{2}} = -\frac{|0>-i|1>}{\sqrt{2}}$

$z|0>=|0>,\ z|1>=-|1>$

So we use X operator to measure $|+>$ or $|->$;

Y operator to measure $\frac{|0>+i|1>}{\sqrt{2}}$ or $\frac{|0>-i|1>}{\sqrt{2}}$; and Z operator to measure $|0>$ or $|1>$.

Let's do one example:

If we want to do Z measurement on qubit one of the following quantum state:

$$\frac{|000>+|111>}{\sqrt{2}}$$

Then, the result will be either $|000>$ or $|111>$ with both 50% chance. The new set of generators will be $\{Z_1I_2I_3,\ I_1X_2I_3, I_1I_2X_3\}$.

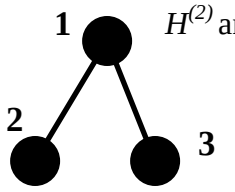
Now, if we want to do X measurement on qubit one of the same quantum state, then:

$$\frac{|000>+|111>}{\sqrt{2}} = \frac{|000>}{\sqrt{2}} + \frac{|111>}{\sqrt{2}}$$
$$= \frac{|+00>+|-00>}{\sqrt{2}} + \frac{|+11>-|-11>}{\sqrt{2}}$$
$$= \frac{|+00>+|+11>}{\sqrt{2}} + \frac{|-00>-|-11>}{\sqrt{2}}$$

Then, the measurement will be either $\frac{|+00>+|+11>}{\sqrt{2}}$ or $\frac{|-00>-|-11>}{\sqrt{2}}$ with 50% chance each. The new set of generators will be $\{I_1Z_2Z_3,\ I_1X_2X_3,\ X_1I_2I_3\}$.

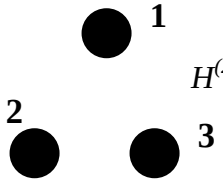
But same as before, we don't want to transfer it back to the standard quantum state representation. So we should find a way to directly apply to our graph states.

Now let's try to graph the first example:
Before Z-Measurement on qubit one:



$H^{(2)}$ and $H^{(3)}$ $\frac{|000>+|111>}{\sqrt{2}}$:

After Z-Measurement on qubit one:



$H^{(2)}$ and $H^{(3)}$ $|000>$ or $|111>$:

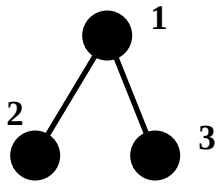
Here are some rules to follow (if we want to do measurement on qubit a: *G* is the vector space of the graph state, and *E*(a, b) means point *a* and point *b* are connected):

Measurement	***HOW***
X	**1.** Pick a neighbor b **2.** Invert edges in between b and $N_a\backslash\{b\}$ **3.** Invert edges in the set $N_a \cap N_b$ **4.** Invert edges between N_a and N_b
Y	***G − E(N$_a$,N$_a$):*** Invert N_a N_a={b:*E*(a,b)}
Z	***G − {a}:*** Remove vertex a and all its edges.

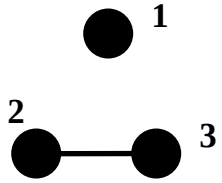
Let's follow the rule and do another example:

We want to do y-measurement on qubit one of the following graph state:

Before Y-Measurement on qubit one:



After Y-Measurement on qubit one (inverting $N_{a)}$:



V. References:

[1] Nielsen, Michael and Chuang, Isaac. *Quantum Computation and Quantum Information*. Cambridge University Press: 2000.

[2] Bacon, Dave. Professor at California Institute of Technology. *Personal interview*. Pasadena, CA: 2004.

[3] Aliferis, Panos. Graduated student at California Institute of Technology. *Personal Interview*. Pasadena, CA: 2004.

[4] M. Hein, J. Eisert, H.J. Briegel. *Multi-party entanglement in graph state.* ARXIV.ORG, July 10, 2004

[5] D. Gottesman. *Stabilizer Codes and Quantum Error Correction*. ARXIV.ORG, May 28, 1997.

[6] J. Preskill. *Physics 229 Lecture Notes*, chapter 7

[7] D. Fattal, T. Cubitt, Y. Yamamoto, S. Bravyi, and I. Chuang. *Entanglement in the stabilizer formalism.* ARXIV.ORG, June 23, 2004