The Gottesman-Knill Theorem

What is it and what does it mean?

Nate Stemen (he/they)

9/12/2020

QIC 710 Final Project

The Gottesman-Knill Theorem

Theorem ([Gottesman, 1998])

*A quantum circuit using only the following elements can be efficiently **simulated** on a classical computer:*

1. *Qubits prepared in computational basis states*
2. *Quantum gates from the **Clifford group***
3. *Measurements in the computational basis*

What does a classical simulation of a quantum computer mean?

Two different main kinds of simulation possible:

What does a classical simulation of a quantum computer mean?

Two different main kinds of simulation possible:

Strong Simulation

Given an input x to our quantum computer, compute $p(x)$.

What does a classical simulation of a quantum computer mean?

Two different main kinds of simulation possible:

Strong Simulation

Given an input x to our quantum computer, compute $p(x)$.

Weak Simulation

Given an input x, compute a sample from $p(x)$.

What does a classical simulation of a quantum computer mean?

Two different main kinds of simulation possible:

Strong Simulation

Given an input x to our quantum computer, compute $p(x)$.

Weak Simulation

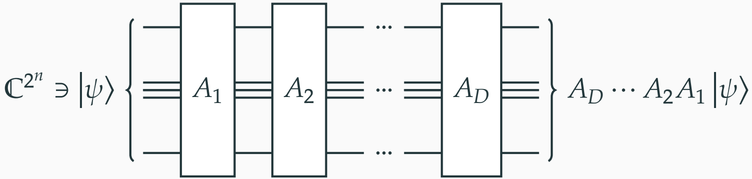
Given an input x, compute a sample from $p(x)$.

Gottesman-Knill theorem deals with weak simulation.

Strong simulation of quantum computers shown to be #**P**-hard [Nest, 2010].

How can we (naïvely) simulate a quantum computer?

Suppose we have n qubits and we want to run them through D gates.



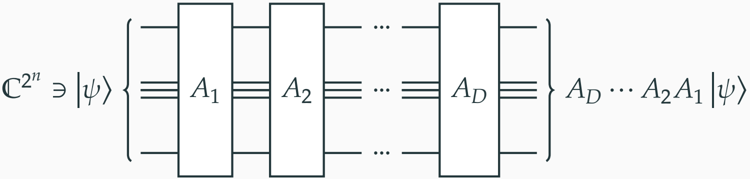
Final state contains $D-1$ matrix multiplications, each costing $O\left(2^{3n}\right)$[1], so total cost is $O\left(D2^{3n}\right)$.

Simulating Grover's algorithm on 40 qubits took nearly a full day! [Viamontes et al., 2004]

[1] Theoretically possible to get $O\left(2^{2.373n}\right)$.

How can we (naïvely) simulate a quantum computer?

Suppose we have n qubits and we want to run them through D gates.



What if we restrict the gates A_i?

[1] Theoretically possible to get $O\left(2^{2.373n}\right)$.

Stabilizer Formalism

- Let $\left|\psi\right\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$

Stabilizer Formalism

- Let $\left|\psi\right\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$
- $X \otimes X \left|\psi\right\rangle = \left|\psi\right\rangle$ and $Z \otimes Z \left|\psi\right\rangle = \left|\psi\right\rangle$

Stabilizer Formalism

- Let $\left|\psi\right\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$
- $X \otimes X \left|\psi\right\rangle = \left|\psi\right\rangle$ and $Z \otimes Z \left|\psi\right\rangle = \left|\psi\right\rangle$
- $\left|\psi\right\rangle$ is the *unique* state stabilized by both of these operators.

Stabilizer Formalism

- Let $\left|\psi\right\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$
- $X \otimes X \left|\psi\right\rangle = \left|\psi\right\rangle$ and $Z \otimes Z \left|\psi\right\rangle = \left|\psi\right\rangle$
- $\left|\psi\right\rangle$ is the *unique* state stabilized by both of these operators.
- This hints at the possibility of describing some states not as vectors in $\mathbb{C}^{2^n}$, but of operators.

Pauli Group

Let X, Y, Z denote the standard single-qubit Pauli operators:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \qquad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

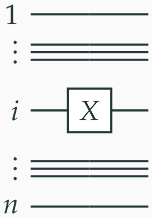
Pauli Group

Let X, Y, Z denote the standard single-qubit Pauli operators:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad Y = \begin{pmatrix} 0 & -\mathrm{i} \\ \mathrm{i} & 0 \end{pmatrix} \qquad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Take X_i, Y_i, Z_i to denote X, Y and Z acting on the i-th qubit, and with the identity everywhere else.

$$X_i := \mathbb{1} \otimes \cdots \otimes \overbrace{X}^{i\text{th operator}} \otimes \cdots \otimes \mathbb{1}$$



Pauli Group

Let X, Y, Z denote the standard single-qubit Pauli operators:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad Y = \begin{pmatrix} 0 & -\mathrm{i} \\ \mathrm{i} & 0 \end{pmatrix} \qquad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Take X_i, Y_i, Z_i to denote X, Y and Z acting on the i-th qubit, and with the identity everywhere else.

$$P_n := \{\pm\mathbb{1}, \pm\mathrm{i}\mathbb{1}, \pm A_i, \pm\mathrm{i}A_i : A_i \in \{\mathbf{1}, X_i, Y_i, Z_i\}\} \equiv \langle X_i, Z_i \rangle$$

- P_n forms a group under matrix multiplication.
- Every pair of elements either commute or anti-commute.
- $|P_n| = 4 \cdot 4^n$

Stabilizer States

Let S be a subgroup of P_n. Define the *vector space* V_S as the states stabilized by everything in S.

$$V_S := \left\{ \left|\psi\right\rangle \in \mathbb{C}^{2^n} : g\left|\psi\right\rangle = \left|\psi\right\rangle, \forall g \in S \right\}$$

Example

Take P_3 and subgroup $S = \{\mathbb{1}, Z_1Z_2, Z_2Z_3, Z_1Z_3\}$. Note that $|000\rangle, |001\rangle, |110\rangle, |111\rangle$ are stabilized by Z_1Z_2, and $|000\rangle, |100\rangle, |011\rangle, |111\rangle$ are stabilized by Z_2Z_3. These, together with the fact that $Z_1Z_3 = (Z_1Z_2)(Z_2Z_3)$ tell us that $V_S = \{|000\rangle, |111\rangle\}$. In this case we can write $S = \langle Z_1Z_2, Z_2Z_3\rangle$.

Stabilizer States

Let S be a subgroup of P_n. Define the *vector space* V_S as the states stabilized by everything in S.

$$V_S := \left\{ \left|\psi\right\rangle \in \mathbb{C}^{2^n} : g\left|\psi\right\rangle = \left|\psi\right\rangle, \forall g \in S \right\}$$

Example

Take P_3 and subgroup $S = \{\mathbb{1}, Z_1Z_2, Z_2Z_3, Z_1Z_3\}$. Note that $|000\rangle, |001\rangle, |110\rangle, |111\rangle$ are stabilized by Z_1Z_2, and $|000\rangle, |100\rangle, |011\rangle, |111\rangle$ are stabilized by Z_2Z_3. These, together with the fact that $Z_1Z_3 = (Z_1Z_2)(Z_2Z_3)$ tell us that $V_S = \{|000\rangle, |111\rangle\}$. In this case we can write $S = \langle Z_1Z_2, Z_2Z_3\rangle$.

S and V_S uniquely determine each other!

Stabilizer States

Let S be a subgroup of P_n. Define the *vector space* V_S as the states stabilized by everything in S.

$$V_S := \left\{ \left|\psi\right\rangle \in \mathbb{C}^{2^n} : g\left|\psi\right\rangle = \left|\psi\right\rangle, \forall g \in S \right\}$$

Example

Take P_3 and subgroup $S = \{\mathbb{1}, Z_1Z_2, Z_2Z_3, Z_1Z_3\}$. Note that $|000\rangle, |001\rangle, |110\rangle, |111\rangle$ are stabilized by Z_1Z_2, and $|000\rangle, |100\rangle, |011\rangle, |111\rangle$ are stabilized by Z_2Z_3. These, together with the fact that $Z_1Z_3 = (Z_1Z_2)(Z_2Z_3)$ tell us that $V_S = \{|000\rangle, |111\rangle\}$. In this case we can write $S = \langle Z_1Z_2, Z_2Z_3\rangle$.

- S must be Abelian
- $-\mathbb{1} \notin S$
- $|S| = 2^{n-k}$ for some $k < n$

What happens when we want to apply a gate?

Let U by an arbitary unitary gate from $\mathbf{U}(2^n)$, $|\psi\rangle \in V_S$ and $g \in S$.

What happens when we want to apply a gate?

Let U by an arbitary unitary gate from $\mathbf{U}(2^n)$, $|\psi\rangle \in V_S$ and $g \in S$.

$$U|\psi\rangle = Ug|\psi\rangle = UgU^\dagger U|\psi\rangle = g'U|\psi\rangle$$

What happens when we want to apply a gate?

Let U by an arbitary unitary gate from $\mathbf{U}(2^n)$, $|\psi\rangle \in V_S$ and $g \in S$.

$$U|\psi\rangle = Ug|\psi\rangle = UgU^\dagger U|\psi\rangle = g'U|\psi\rangle$$

So $U|\psi\rangle$ is stabilized by $UgU^\dagger$, and in general UV_S is stabilized by $USU^\dagger = \left\{UgU^\dagger : g \in S\right\}$.

Corollary

If S is generated by $g_1, \dots, g_n$, then $USU^\dagger$ is generated by $Ug_1U^\dagger, \dots, Ug_nU^\dagger$.

What happens when we want to apply a gate?

Let U by an arbitary unitary gate from $\mathbf{U}(2^n)$, $|\psi\rangle \in V_S$ and $g \in S$.

$$U|\psi\rangle = Ug|\psi\rangle = UgU^\dagger U|\psi\rangle = g'U|\psi\rangle$$

So $U|\psi\rangle$ is stabilized by $UgU^\dagger$, and in general UV_S is stabilized by $USU^\dagger = \{UgU^\dagger : g \in S\}$.

Corollary

If S is generated by $g_1, \ldots, g_n$, then $USU^\dagger$ is generated by $Ug_1U^\dagger, \ldots, Ug_nU^\dagger$.

If G is an Abelian group with $|G|$ the number of elements in G, then the number of generators of G is bounded by $\log_2 |G|$. In particular the number of generators is bounded above by n.

What happens when we want to apply a gate?

Let U by an arbitary unitary gate from $\mathbf{U}(2^n)$, $|\psi\rangle \in V_S$ and $g \in S$.

$$U|\psi\rangle = Ug|\psi\rangle = UgU^\dagger U|\psi\rangle = g'U|\psi\rangle$$

So $U|\psi\rangle$ is stabilized by $UgU^\dagger$, and in general UV_S is stabilized by $USU^\dagger = \{UgU^\dagger : g \in S\}$.

Corollary

If S is generated by $g_1, \dots, g_n$, then $USU^\dagger$ is generated by $Ug_1U^\dagger, \dots, Ug_nU^\dagger$.

If G is an Abelian group with $|G|$ the number of elements in G, then the number of generators of G is bounded by $\log_2 |G|$. In particular the number of generators is bounded above by n.

What if $UgU^\dagger$ doesn't land back in P_n?

What if $UgU^\dagger$ doesn't land back in P_n?

Theorem

Suppose U in any unitary on n qubits with the property that for $g \in P_n$ we have $UgU^\dagger \in P_n$. Then U can be composed from $O(n^2)$ Hadamard, CNOT, and phase gates.

What if $UgU^\dagger$ doesn't land back in P_n?

Theorem

Suppose U in any unitary on n qubits with the property that for $g \in P_n$ we have $UgU^\dagger \in P_n$. Then U can be composed from $O(n^2)$ Hadamard, CNOT, and phase gates.

Definition

The *Clifford Group* is defined to be the set of operators that leave Pauli operators invariant under conjugation.

$$C_n := \left\{ V \in \mathbf{U}(2^n) : VP_nV^\dagger = P_n \right\}$$

Clifford Group

Theorem

Suppose U in any unitary on n qubits with the property that for $g \in P_n$ we have $UgU^\dagger \in P_n$. Then U can be composed from $O(n^2)$ Hadamard, CNOT, and phase gates.

Definition

The *Clifford Group* is defined to be the set of operators that leave Pauli operators invariant under conjugation.

$$C_n := \left\{ V \in \mathbf{U}(2^n) : V P_n V^\dagger = P_n \right\}$$

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \qquad P = \begin{pmatrix} 1 & 0 \\ 0 & \mathrm{i} \end{pmatrix}$$

CNOT

Recap

1. Simulating a quantum computer in general is *really* hard!

Recap

1. Simulating a quantum computer in general is *really* hard!
2. What can we simulate more easily?

Recap

1. Simulating a quantum computer in general is *really* hard!
2. What can we simulate more easily?
3. Stabilizer formalism gave us a way to track operators instead of state vectors (duality between subgroup S of Paulis and vector space of stabilised states V_S)

Recap

1. Simulating a quantum computer in general is *really* hard!
2. What can we simulate more easily?
3. Stabilizer formalism gave us a way to track operators instead of state vectors (duality between subgroup S of Paulis and vector space of stabilised states V_S)
4. Keeping track of the generators of a stabilizer S provide a succinct way to understand how S is changing ($\log |S|$ generators)

Recap

1. Simulating a quantum computer in general is *really* hard!
2. What can we simulate more easily?
3. Stabilizer formalism gave us a way to track operators instead of state vectors (duality between subgroup S of Paulis and vector space of stabilised states V_S)
4. Keeping track of the generators of a stabilizer S provide a succinct way to understand how S is changing ($\log |S|$ generators)
5. Found that elements of the Clifford group can efficiently build elements that conjugate the Pauli group back to the Pauli group

Back to the Theorem

Theorem ([Gottesman, 1998])

A quantum circuit using only the following elements can be efficiently simulated on a classical computer:

1. *Qubits prepared in computational basis states*
2. *Quantum gates from the Clifford group*
3. *Measurements in the computational basis*

- Take $\left|\psi\right\rangle = |0\rangle^{\otimes n}$. Now we can say $S = \langle Z_1, \dots, Z_n \rangle$.
- Under some action $U \in C_n$ state will evolve to $U\left|\psi\right\rangle = UgU^{\dagger}U\left|\psi\right\rangle$ for $g \in S$
- Switch over to describing the change in generators of S
- Need to compute $UZ_1U^{\dagger}, \dots, UZ_nU^{\dagger}$

Back to the Theorem

Recap

We have $|\psi\rangle = |0\rangle^{\otimes n}$, $S = \langle Z_1, \dots, Z_n \rangle$, and $U \in C_n$. We know that $U|\psi\rangle = g'U|\psi\rangle$, so in order to figure out where it evolves to, need to compute the new generators of the space UV_S which are $UZ_1U^\dagger, \dots, UZ_nU^\dagger$.

- Only takes $2n+1$ bits to keep track of a generator: 2 for the n Pauli generators, and 1 for the phase of ± 1
- Specifying $|\psi\rangle$ requires all n generators, so $n(2n+1)$ bits
- Updating the generators only takes $O(n)$
- Total cost $O(n^2)$

X_1	1
X_2	0
X_3	1
Z_1	0
Z_2	0
Z_3	1
± 1	0

Table 1: Encoding $X_1X_3Z_3$

What does this mean?

What makes quantum computers powerful?

What does this mean?

What makes quantum computers powerful?

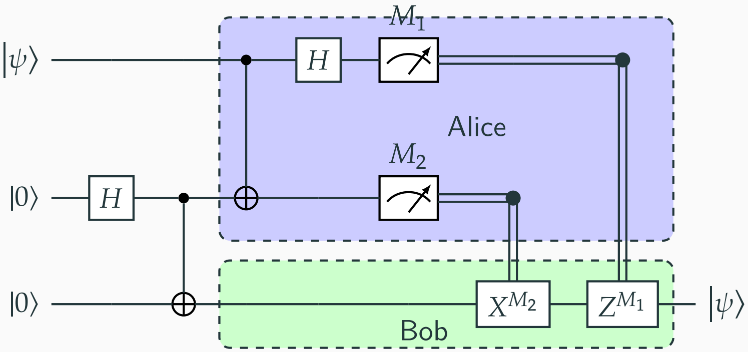
Not *just* entanglement!

What does this mean?

What makes quantum computers powerful?

Not *just* entanglement!

Quantum teleportation

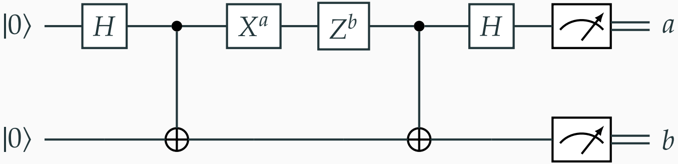


What does this mean?

What makes quantum computers powerful?

Not *just* entanglement!

Superdense coding

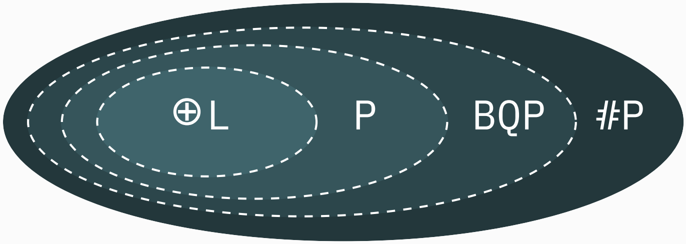


Where do we go from here?

- Clifford gates aren’t enough for universal quantum computation

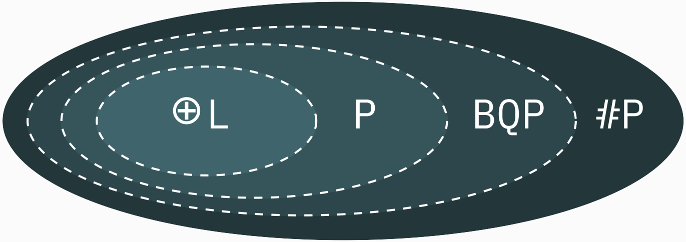
Where do we go from here?

- Clifford gates aren't enough for universal quantum computation
- Clifford group shown to be ⊕**L**-complete [Aaronson and Gottesman, 2004]



Where do we go from here?

- Clifford gates aren't enough for universal quantum computation
- Clifford group shown to be ⊕**L**-complete [Aaronson and Gottesman, 2004]
- Adding *any* 1 or 2-qubit gate[2] will turn the Cliffords into a universal set [Shi, 2002]



[2]That doesn't map computational basis states to computational basis states

Conclusion

Assuming

$\mathbf{BQP} \neq \mathbf{P} \neq \oplus\mathbf{L}$

Strong simulation of quantum computers is *really* hard.

Clifford group is only capable of solving relatively easy problems (both from classical and quantum POV).

Quantum entanglement is not the only contributing factor to the power of quantum computers!

References i

Aaronson, S. and Gottesman, D. (2004).
Improved simulation of stabilizer circuits.
Phys. Rev. A, 70:052328.

Bravyi, S. and Kitaev, A. (2004).
Universal quantum computation with ideal clifford gates and noisy ancillas.
Physical Review A, 71.

Cuffaro, M. E. (2015).
On the Significance of the Gottesman–Knill Theorem.
The British Journal for the Philosophy of Science, 68(1):91–121.

Gottesman, D. (1998).
The Heisenberg Representation of Quantum Computers.
arXiv e-prints, pages quant–ph/9807006.

Nest, M. (2010).
Classical simulation of quantum computation, the gottesmann-knill theorem, and slightly beyond.
Quantum Information & Computation, 10:258–271.

Shi, Y. (2002).
Both toffoli and controlled-not need little help to do universal quantum computation.
Quantum Information and Computation, 3.

Viamontes, G., Markov, I., and Hayes, J. (2004).
Improving gate-level simulation of quantum circuits.
Quantum Information Processing, 2.

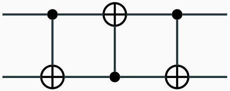
☺**Thank you!**☺

Questions?

Example

Alice's Broken Quantum Computer ☹

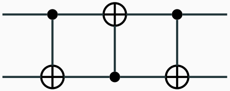
Alice's quantum computer is working too well. Instead of performing single controlled-NOT gates, it does three at a time. What is it actually doing?



Example

Alice's Broken Quantum Computer ☹

Alice's quantum computer is working too well. Instead of performing single controlled-NOT gates, it does three at a time. What is it actually doing?



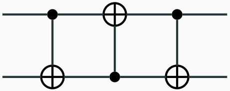
Because X_1, X_2, Z_1, Z_2 generate the Pauli group we can follow what happens to them under the evolution of this circuit.

$$X_1 = X \otimes \mathbb{1} \xrightarrow{\text{CNOT 1}} \text{CNOT} \cdot (X \otimes \mathbb{1}) \cdot \text{CNOT}^{\dagger} = X \otimes X$$

Example

Alice's Broken Quantum Computer ☹

Alice's quantum computer is working too well. Instead of performing single controlled-NOT gates, it does three at a time. What is it actually doing?



Because X_1, X_2, Z_1, Z_2 generate the Pauli group we can follow what happens to them under the evolution of this circuit.

$$X_1 = X \otimes \mathbb{1} \xrightarrow{\text{CNOT 1}} X \otimes X \xrightarrow{\text{CNOT 2}} \mathbb{1} \otimes X \xrightarrow{\text{CNOT 3}} \mathbb{1} \otimes X = X_2$$

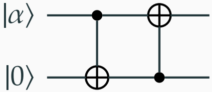
$$Z_1 = Z \otimes \mathbb{1} \xrightarrow{\text{CNOT 1}} Z \otimes \mathbb{1} \xrightarrow{\text{CNOT 2}} Z \otimes Z \xrightarrow{\text{CNOT 3}} \mathbb{1} \otimes Z = Z_2$$

Further, we can show $X_1 \longleftrightarrow X_2$ and $Z_1 \longleftrightarrow Z_2$. This is exactly a swap operation!

Example, continued

Alice's less Broken Quantum Computer ☺

By dint of no little hard work, Alice has partially fixed her quantum computer. Now it only does 2 CNOTs at a time. Unfortunately, she can only get this improvement if she puts a $|0\rangle$ as the second input qubit. What does it do now?



In this case we see the initial state $|\psi_0\rangle = |\alpha\rangle \otimes |0\rangle$ is stabilized by Z_2.

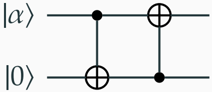
State will always be a +1 eigenvector of Z_2, so it follows that $(Z \otimes \mathbb{1})(Z \otimes Z) = \mathbb{1} \otimes Z$.

Circuit still performs a swap!

Example, continued

Alice's less Broken Quantum Computer ☺

By dint of no little hard work, Alice has partially fixed her quantum computer. Now it only does 2 CNOTs at a time. Unfortunately, she can only get this improvement if she puts a $|0\rangle$ as the second input qubit. What does it do now?



In this case we see the initial state $|\psi_0\rangle = |\alpha\rangle \otimes |0\rangle$ is stabilized by Z_2.

$$X_1 = X \otimes \mathbb{1} \xrightarrow{\text{CNOT 1}} X \otimes X \xrightarrow{\text{CNOT 2}} \mathbb{1} \otimes X$$

$$Z_1 = Z \otimes \mathbb{1} \xrightarrow{\text{CNOT 1}} Z \otimes \mathbb{1} \xrightarrow{\text{CNOT 2}} Z \otimes Z$$

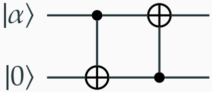
$$Z_2 = \mathbb{1} \otimes Z \xrightarrow{\text{CNOT 1}} Z \otimes Z \xrightarrow{\text{CNOT 2}} Z \otimes \mathbb{1}$$

State will always be a +1 eigenvector of Z_2, so it follows that $(Z \otimes \mathbb{1})(Z \otimes Z) = \mathbb{1} \otimes Z$

Example, continued

Alice's less Broken Quantum Computer ☺

By dint of no little hard work, Alice has partially fixed her quantum computer. Now it only does 2 CNOTs at a time. Unfortunately, she can only get this improvement if she puts a $|0\rangle$ as the second input qubit. What does it do now?



In this case we see the initial state $\left|\psi_0\right\rangle = |\alpha\rangle \otimes |0\rangle$ is stabilized by Z_2.

$$X_1 \longrightarrow \mathbb{1} \otimes X \qquad Z_1 \longrightarrow Z \otimes Z \qquad Z_2 \longrightarrow Z \otimes \mathbb{1}$$

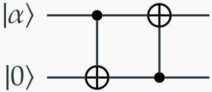
State will always be a +1 eigenvector of Z_2, so it follows that $(Z \otimes \mathbb{1})(Z \otimes Z) = \mathbb{1} \otimes Z$.

Circuit still performs a swap!

Example, continued

Alice's less Broken Quantum Computer ☺

By dint of no little hard work, Alice has partially fixed her quantum computer. Now it only does 2 CNOTs at a time. Unfortunately, she can only get this improvement if she puts a $|0\rangle$ as the second input qubit. What does it do now?



In this case we see the initial state $|\psi_0\rangle = |\alpha\rangle \otimes |0\rangle$ is stabilized by Z_2.

$$X_1 \longrightarrow \mathbb{1} \otimes X \qquad Z_1 \longrightarrow Z \otimes Z \qquad Z_2 \longrightarrow Z \otimes \mathbb{1}$$

State will always be a +1 eigenvector of Z_2, so it follows that $(Z \otimes \mathbb{1})(Z \otimes Z) = \mathbb{1} \otimes Z$.

$$X_1 \longrightarrow \mathbb{1} \otimes X \qquad Z_1 \longrightarrow \mathbb{1} \otimes Z$$

Circuit still performs a swap!