Quantum Error Correction - Lecture 6

with Dan Browne

March 28, 2016

Contents

Recall that we spoke about fault tolerant constructions. We want each component failure which occurs with probability p to cause up to c single errors in each code block.

We must now consider:

- Unitary gates
- State preparation
- Measurements

The problem is that many constructions are based on CNOT gates. In this general error model, a CNOTfailing can cause a 2-qubit error.

A CNOT inside a code block fails to meet the conditions of fault-tolerance. We can absolutely not have any instance of a CNOT gate inside a code block.

So, we allow entangling gates on between code blocks. We call these **transversal gates**. A unitary gate is transversal if it contains no entangling gates within a code block. Let us look at the Steane code as an example.

For a code block with seven qubits, and logical operators $\bar{X} = X^{\otimes 7}$ and $\bar{Z} = Z^{\otimes 7}$. These do not entangle. If $\bar{X}$ and $\bar{Z}$ are transversal, then $\bar{Y}$ is too since $\bar{Y} = i\bar{X}\bar{Z}$. The Steane code also has $\bar{H} = H^{\otimes 7}$. and $\bar{S} = (S\dagger)^{\otimes 7}$.

Question: Why must we take the adjoint of the S gate?

However, for the Steane code, it turns out that we can create a CNOT gate that is also transversal!

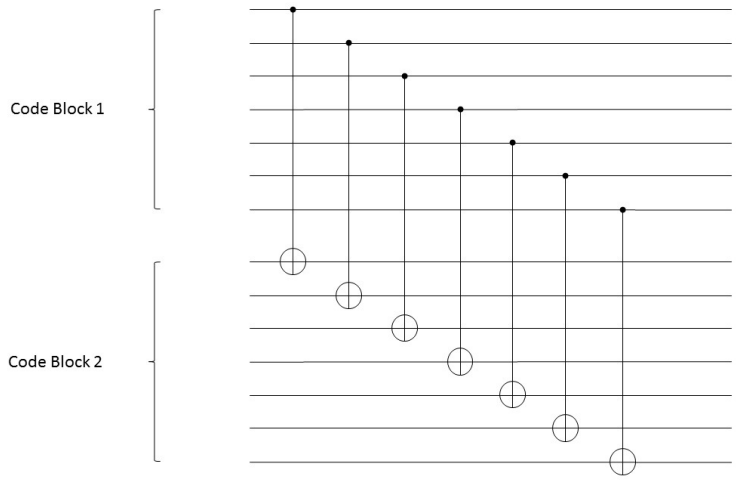

Figure 1: Transversal CNOT gate.

We just have to make sure that every CNOT component acts between code blocks. But later, we will need universal gates. It turns out that the stabiliser formalism and transversal gates is not fault tolerant on its own. But recall that the Clifford group is fault tolerance.

1 Fault tolerant stabiliser measurement

Recall our general Pauli tensor product projector that can be used as a measurement:

$$P = p_1 \otimes P_2 \otimes P_3 \dots P_n \tag{1}$$

Recall the phase kick-back circuit that we wanted. Each projector projects onto a qubit and the outcome is recoded in an ancilla. The

ancilla doesn't need to be in the code block. However, an error on the ancilla means that it can propagate down.

Z-errors would not propagate down, but X errors would after the 1st gate. However, we can fix this with the repetition code. We can create a so-called cat-state. The trick is that for an ancilla, instead of $|+\rangle$, we use $|0\rangle^{\otimes n} + |1\rangle^{\otimes n}$. This is GHZ state, but as mentioned is also called a cat-state. It is a repetition code implementation.

1.1 Example: 3-qubit code

Given the ancilla state $|000\rangle + |111\rangle$. Let us then apply th projectors P_1, P_2 and P_3 on each of the registers. Finally, we distinguish $|000\rangle + |111\rangle$ from $|000\rangle - |111\rangle$. Note that this cat state is a stabiliser state!

It turns out that what before was a weakness, the fact that any single Z qubit gate would flip the entire state, which caused the code to fail at detecting phase errors now becomes the weakness. We only need to flip one of the qubits to change the entire state. However, whereas the 3-qubit repetition code fails at error correcting, we here apply the stabiliser formalism in addition.

For the full measurement to be fault-tolerant, we need a fault-tolerant way to prepare and distinguish between the final state. We can introduce the notion of a **verification circuit**. For the state $|000\rangle + |111\rangle$ we cna measure the stabilisers XXX, ZZI, IZZ. So, the entire circuit would look as follows.

Start with the $|+\rangle$ state in the ancilla. Then use the initisalisation circuit for the repetition code with CNOTs to create $|000\rangle + |111\rangle$. But how can we determine that the preparation succeeded? We create this additional circuit shown below.

Here, the line connected by two dots is the control Z gate. It is symmetric between target and control qubits. We can only use the ancilla if both checks pass. This suppresses multi-qubit errors in the state and keeps single qubit errors of order p.

This is an example of **post-selection**. It is, in a sense, wasteful, but it is effective.

To distinguish between $|000\rangle + |111\rangle$ and $|000\rangle - |111\rangle$, we inverse the circuit. Using the CNOT gates and measuring will give us either $|+\rangle$ or $|-\rangle$, which indicate the final state.

Finally, while the above corrects for single or double errors in the preparation process, it still cannot correct for an XXX error. However, we can easily repress it by repeating the measurement. For each time,

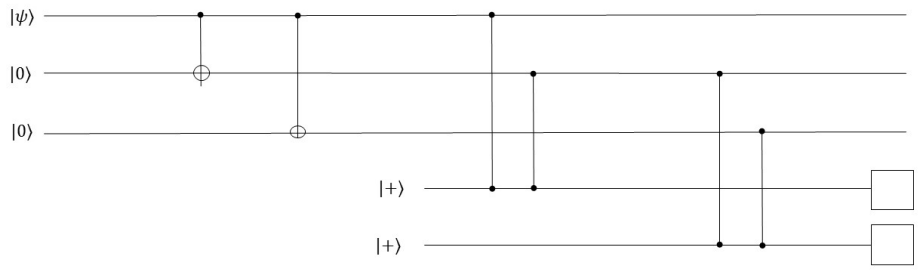

Figure 2: Verification circuit.

we can measure three times and take the majority vote. This suppresses all errors to at least p^2.

So, in summary, measurement errors can be suppressed by repeating the overall procedure.

2 State preparation

Let us have a look at state preparation in fault tolerant measurements of stabiliser generators. For example, we will want a fault-tolerant measurement of $\bar{Z}$.

Any unitary correction that we perform on the state will automatically be transversal. The method we shall look at now is one of Steane's methods. Its prevalence was recently overtaken by topological codes.

Note: for some reason I don't have many notes on this case.

3 Threshold example

There are various things that we need to consider.

- Different architectures and different error assumptions lead us to different thresholds.

- We can have analytic threshold which are rigorously proven, but they tend to underestimate the performance of the code (that, is we will always assume the worst-case scenario.
- Numerical thresholds are obtained by simulations.

3.1 Examples of thresholds

The steane code has an analytic threshold of $p = 10^{-5}$ and an analytic threshold of $p = 10^{-3}$.

Other codes, such as the Bacon-Shor code, using a so-called subsystem code, have an analytic threshold of about 10^{-4}.

The surface code on the other hand has an analytic threshold of 10^{-1}.

4 CSS code

The Steane code is an example of a so-called Calderbank-Steane-Shor code (CSS). The basic idea is that the X and Z are independent. As we saw before, two classical codes are combined – one to correct X errors and one to correct Z errors.

Generators of stabilisers can be found where each generator contains either X and I tensor factors, or Z and I tensor products.

CSS codes derived from this family are called **linear codes**. They are codes where codewords are formed as linear combinations, which means that they are linear in terms of basis codewords.

As an example, we can look at the Hamming code: $(n = 7, k = 4)$. This code encodes the states into

$$|0001\rangle = |0001011\rangle \tag{2}$$

$$|0010\rangle = |0010110\rangle \tag{3}$$

$$|0100\rangle \to |0100101\rangle \tag{4}$$

$$|1000\rangle \to |100011\rangle \tag{5}$$

Here, the three last digits tell us about the parity of the code. Then, any permutation would be

$$|0101\rangle = |0001\rangle + |0100\rangle \to |0101110\rangle \tag{6}$$

where the above uses bitwise addition mod 2. In general, it implies htat

$$abcd \to abcd(a+b+c)(a+c+d)(a+b+d) \tag{7}$$

where the last few terms are indeed parity checks. Each codeword satisfy certainty parity conditions. Errors are detected by violations of these conditions. For example, the lst 3 bits plus the 1st bit of each codeword eaul zero. An error would be detected if these are found to not do so.

For example, we can then encode these states in a matrix:

$$(1000111) \cdot x^T = 0 \tag{8}$$

We can write down parity constraints in a matrix, which is what we have done here. We then have a set of independent parity conditions. This turns out to be equivalent to the stabiliser generators!

In particular, with a slight re-ordering of the bits, we see that the Hamming code has parity check

$$0001111, 0110011, 1010101 \tag{9}$$

So for each line y, we require $y \cdot x^T = 0$. We can check the errors by checking each line. This is where the stabiliser generator come from!

CSS codes are constructed by mapping parity checks for a linear code to X-stabiliser generators and Z-stabiliser generators.

In the Steane code, we had the transversal T gate which is not in the Clifford group. Recall that

$$S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \tag{10}$$

$$S^4 = I \tag{11}$$

The Steane code codewords (see handouts) are very long. The Hamming weight for the states in the superposition is the same but for the first one $|0000000\rangle$. We now want to show that the S gate is transversal.

For the codeword $|0\rangle_L$ we have weights 0 and 4 for the codewords. For the codeword $|1\rangle_L$ we have weight 7 and 3. Thus, if we apply $\bar{S}$,

$$(S^\dagger)^{\otimes 7} |x\rangle = -i)^{weight(x)} |x\rangle \tag{12}$$

Note that i^x obeys mod 4. So then,

$$(-i)^{\otimes 7} |0\rangle_L = i |0\rangle_L \tag{13}$$

$$(S^\dagger)^{\otimes 7} |1\rangle_L = (-1)^{3 \mod 4} |1\rangle_L = i |1\rangle_L \tag{14}$$

Similarly, there exists a 15-qubit code. Is has weight 8 on its codewords. Here,

$$\bar{T} = (T^\dagger)^{\otimes 15} \tag{15}$$

using the same argument. However, there is a bit problem for the 15-qubit code: It does not have a Hadamard gate!

Finally, we can prove that it is impossible for a stabiliser code to have a universal set of gates. This is the **Eustin-Knill Thorem**. The best way to get around this is to use magic state distillation.

5 Magic state distillation

The key idea here is to use **state injection**. Essentially, it will replace the pauli state preparation. We use the state $T|+\rangle$ as a resource to implement T in the Clifford gates and Pauli basis.

You can check the derivation for this yourself!

The question now is how to fault-tolerantly prepare $T|+\rangle$.

One way to do so would be to use fault-tolerant state preparation on the Steane code. This is called the Shor-scheme, see Nielsen and Chuang Section 10.6.2. However, the problem is that we get a very poor threshold.

Here, we shall instead look at the magic distillation method.

The idea is the following: We shall attempt to make $T|+\rangle$ without fault-tolerance. Consider a simple circuit that applies T to the $|+\rangle$ followed by some error. The final state is not $T|+\rangle$ but rather some other mixed state ρ. We don't apply any error correction throughout the circuit.

This error is generally unavoidable. Instead, we use magic state distillations. We take N copies of ρ. We then measure a set of stabiliser generators on all of these states, then decode it to get ρ'.

We can show that provided that ρ is close enough to $T|+\rangle\langle+|T^\dagger$, ρ is good enough after distillation.

This works for every code but the 15 qubit Reid-Miller-Code. Check Kitaev 2004 for a proof. They use the property that the T-gate is transversal for the 15 qubit code.

These states are important because

- $T|+\rangle$ enables us to achieve universal quantum computing with the Clifford gates.
- Magic state distillation requires only Clifford group gates!

Aside: Recall that stabiliser states form an octahedron in the Bloch Sphere. The this means tht we can convex closure of all points inside the octahedron. Just like we cannot obtain a pure state by convex

combination of two mixed states, we cannot create $T|+\rangle$ from states inside the octahedron - $T|+\rangle$ lies outside it!

We define the fidelity as

$$F = \sqrt{\langle\psi|\,\rho\,|\psi\rangle} \tag{16}$$

where

$$|\psi\rangle = |\Psi\rangle \tag{17}$$

Then, define

$$1 - F^2 = \epsilon \tag{18}$$

which is the distance between the obtained state ρ and the desired state ψ. Magic state distillation works really well for the lower thresholds. See some Jupyter notebook notes with a numerical calculation on the Moodle website.