Master's Thesis

Emma Storberg

October 26, 2025

Contents

Chapter 1

Introduction

Why is this an interesting topic?
Use some of project description here.

Why quantum computers over classical?

What do we need for a quantum computer to work?
To build a quantum machine in practice, we need a large number of qubits. The root of the issue is that quantum informaiton is extremely fragile. You can literally ruin it just by looking at it. Qubits are unstable and very vulnerable to noise that changes the value the qubit holds. We therefore wish to construct circuits of multiple qubits that work together to protect the information from errors, such that they form a single logical qubit that can have its errors located and corrected, making them more stable. The standard method of error correction that is used in all existing quantum computers (find source for this) is the *surface code*.

Explain surface code here.

What am I going to do?
As a starting point for this project, I aim to recreate the work of Samuel Jacques, as laid out in his 2024 talk.

Jacques takes a very pessimistic view on the potential use cases for Grover's algorithm to break AES, by which I mean he believes these use cases are entirely nonexistent. His argument relies on a series of assumptions for this to be the case. In the beginning stages of this project, I will tackle these assumptions case by case and see if it changes the conclusion at all. Along the way, we will run into a number of topics that I am not really familiar with, and we will have to acquire some knowledge to say if the arguments presented by Jacques still hold in these alternative cases.

For instance, if we work other error correcting codes like *BB codes* or *tile codes*, does that change the picture?

Chapter 2

Background

2.1 What is Grover's algortihm?

Include content from TEK5550 text on Grover's algorithm here.

2.2 What is Jacques' full line of argumentation?

In the normal case when we would use Grover's algorithm, we assume no structure. The argument for there being security in this case is that the number of potential keys is so large that we would not be able to do a brute-force search attack on a classic computer in any reasonable amoutn of time (brute-force being the only option here beceause there is no structure to the problem or the way the keys are determined). The circuit for the quantum algorithm Grover's is such that we may have a speedup on the order of the square root. We also know (from my own previous work) that the runtime of Grover's scales with the number of Grover iterates, which themselves scale with the number of queries to the arbitrary funciton f, or the AES algorithm in our case, which is needed to see if a binary string x is a solution. Each Grover iteration requires running the entire AES encryption circuit, and this is the costly part.

There might also be constant factors here we don't know about, so to see the full picture, we need to know exactly how our f, namely AES, will be implemented in a circuit. Jacques provides a table of estimates of different values on page 20 of the PDF (slide 8 in the deck). The three quantities of apparent interest are T gates (simple, computationally easy gates), Clifford gates (a different type of gate that requires a huge number of qubits) and the depth, which I take to mean the number of gates that have to be applied serially, kind of functioning as a proxy for time (as they cannot be run in parallel). Finally, we have an estimate of a couple thousand logical qubits required for each size of AES in the table, but of course there could be millions of qubits hiding behind this number.

We definitely need some form of error correction, since quantum computations are very prone to noise. But we are orders of magnitude off the number of physical qubits we need in order to be anywhere near the estimated number of necessary logical qubits.

At this point, he introduces NIST's 2017 MAXDEPTH metric as a baseline for how many logical gates/operations the current quantum computing architectures perform serially over certain periods of time, like a year (2^{40}), a decade (2^{64}) or a millenium ($2^{9}6$)[1]. Additionally, he makes clear that any sort of realistic attack on AES would have to be parallelized. However, Grover doesn't parallelize well, as additional partitions add an overhead every time (simple calculation/explanation on slide 33 in the deck). We could argue that since the classical cost is on the order of 2^{128} for AES-128 and Grover offers a square root speedup, the new cost will be 2^{64}, with some small constant to account for overhead and setting up the quantum circuit. However, we have now shown that the bad parallelism of Grover means it is not on the order of 2^{64}, and we have no idea what the constant will be if we use a different type of code other than surface codes.

This paper will attempt to tackle just that. What happens if we use something else instead, and what might that be?

2.3 Assumptions

In this section, I will highlight a few different assumptions that Jacques makes. If we get rid of them, that might be a possible avenue of improving his prognosis.

2.3.1 The attack is completely structureless

What if the attack has more structure?
I think this is not the best path to go down as I believe this fundamentally alters the problem away from unstructured search.

2.3.2 We use surface codes as a form of error correction at the current detection rate.

Thomas has shown me a different type of promising error correction known as LDPC (Low Density Parity Check) codes, including BB codes and tile codes. These codes have a number of advantages over surface codes, such as a higher threshold for error correction and lower qubit overhead.

Why (or how) might this change the conclusion? Where in the argumentation does the assumption of surface codes have an impact?

[1]He also points out that this limit does not reflect decoherence concerns, i.e. the quantum state collapsing and quantum data disappearing.

I think investigating this is a good place to start.

2.3.3 We encode the Grover circuit with Clifford + T gates

This seems related to a really fundamental aspect of his argument, since the number of T-gates is what ultimately determines the time complexity of the algorithm. If there was a better way to create the circuit that requires less T-gates (or T-gate equivalents), we wouldn't be able to rely on his estimate, since the huge cost of the T-gates as he assumed is part of why this works out so poorly.

2.4 Surface Code

This writeup[2] explains surface code pretty well. May be worth testing out their Jupyter notebook too.

Surface code works by creating a lattice of data qubits and measurement qubits interspersed. Question: Where does the ratio "1000:1" come from? Seem like we need 4 per qubit????

2.5 Alternatives to Surface Code

2.5.1 Bivariate Bicycle Code

Bivariate Bicycle code, also known as BB code, is a quantum code in the Low Density Parity Check (LDPC) code class. Though similar to surface codes at a high level of abstraction, BB codes differ in that their check operators are not geometrically local, and has therefore been called "surface code with extra long distance checks" (summer student project, p. 6).

It is formally defined as a pair of binary matrices that are sums of shift matrices to integer powers.

Python code for simulating BB codes can be found here: `https://github.com/AntonBrekke/BBCODE-QEC-2025/tree/main`.

2.5.2 Tile Code

Tile code somewhat resembles the toric surface code, and are also related to BB codes. However, it does not have periodic boundary conditions. It has the stabilizers located outside of a "tile" of qubits, instead of interspersed between.

They have created an algorithm which helps you select which qubits to check (?).

[2]`https://github.com/mcclow12/Quantum-Computing-Surface-Code-Simulation/blob/master/writeup.pdf`

Chapter 3

Methods

Describe your computational physics methods here.

Chapter 4

Results

Show figures and tables from your experiments here.

Chapter 5

Conclusion

Summarize your findings and future work.