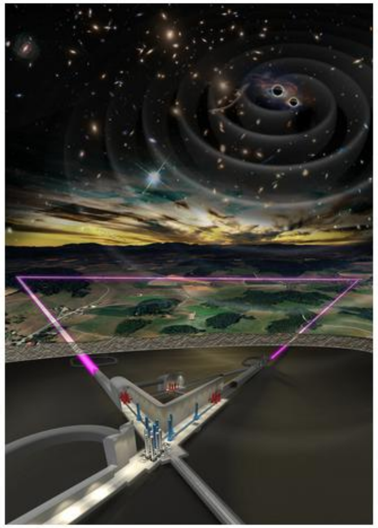
Applications of Quantum Computing to GW data analysis: Progress Update

Table of Contents

New Generation Detectors and Data Collection



3G Detectors: Einstein Telescope

- Order of magnitude better amplitude sensitivity
- Detection of frequencies as low as 1 Hz
- Allows to make observations from the whole visible universe, all the wayto its dark age

Massive increase in the amount of the data to be analysed

So much in fact that modern data-analysis techniques and supercomputer are not enough for 3G detectors (Bird, 2019).

Possible solution: Quantum computing

Collaboration Efforts

Expertise in Gravitational Wave Research

Utrecht University

- Practical experience in data-analysis aspects
- Matched Filtering
- Parameter estimation

Quantum Technology

- Rydberg Atom Quantum Computer
- Practical expertise in QC
- Theoretical expertise in QC

Ideas for possible applications of QC in GW data analysis

- Parameter Estimation (VQE?)
- Matched Filtering (Grover's Algorithm)
- Machine Learning Models
- SVD (VQE)
- and more that we haven't thought of yet...

Grovers Algorithm

It is a quantum search algorithm. The easiest example for this is a quantum register that has been fully put to equal superposition. This is done by applying Hadamard gates to each qbit.

$$H|0\rangle H|0\rangle = |++\rangle = \left[\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right] = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)$$

Some of the computational bases (the bases created by the combination of all the qbits). Will somehow match your search, while the others don't.

The algorithm then has two parts:

1. An oracle that sifts the global phase of the solutions (no overall change if measured)
2. A diffusion operator that will transfer some of the amplitude of the non-solutions to the solutions.

Can be though of graphically:

- all the solution bases are combined together to form basis $|\omega\rangle$
- all the non-solutions create the orthonormal basis $|\omega_\perp\rangle$

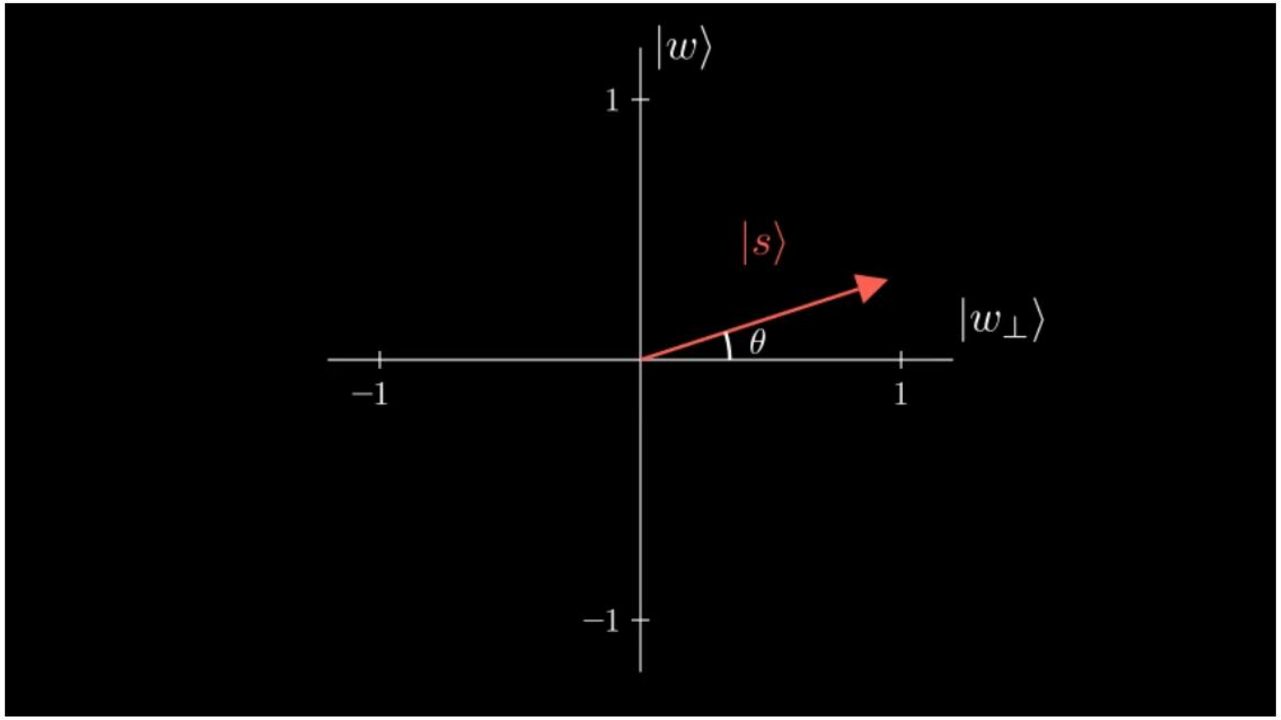
We can then write the state as

$$|s\rangle = \sqrt{\frac{r}{N}}\,|w\rangle + \sqrt{\frac{N-r}{N}}\,|w_\perp\rangle$$

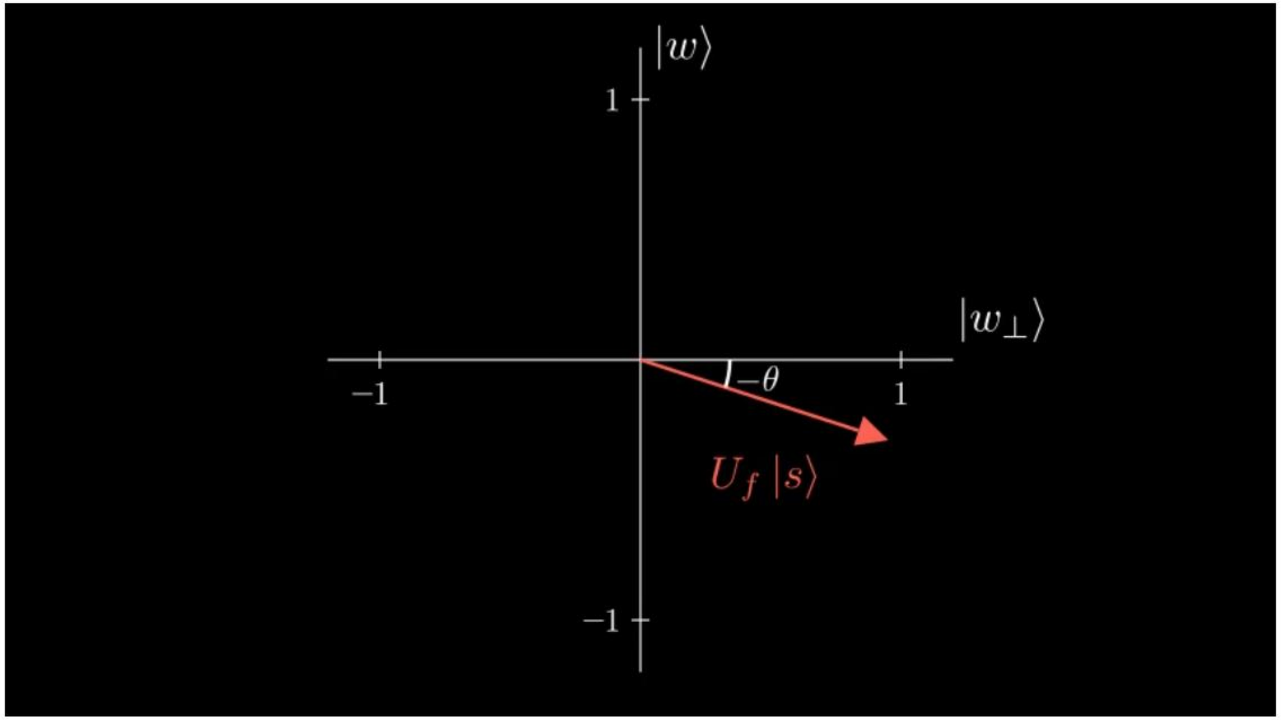
with r being the number of solutions and N being the number of elements.

The the oracle simply gives a negative sign to the first element and the diffusion operator reflects it on $|s\rangle$. This results in a state vector closer to $|w\rangle$, meaning the prob. of measuring a correct solution increases.

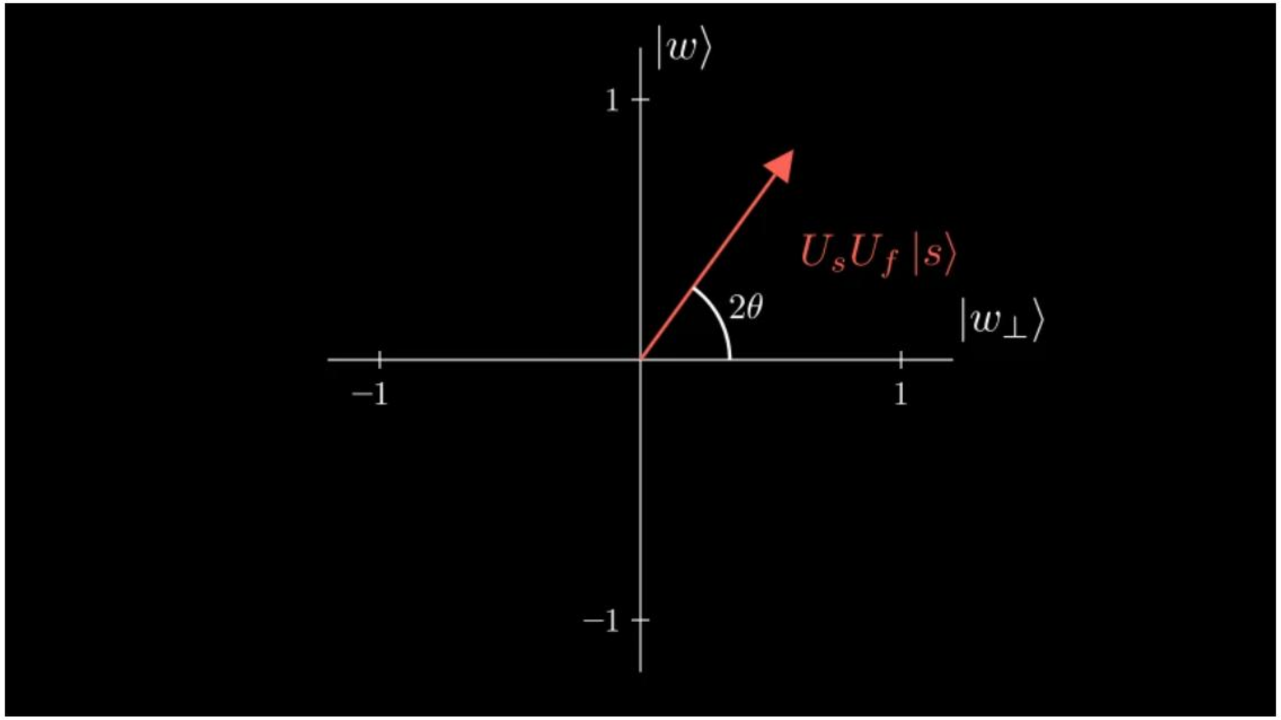
Out[48]:



Out[48]:



Out[48]:



Quantum Counting

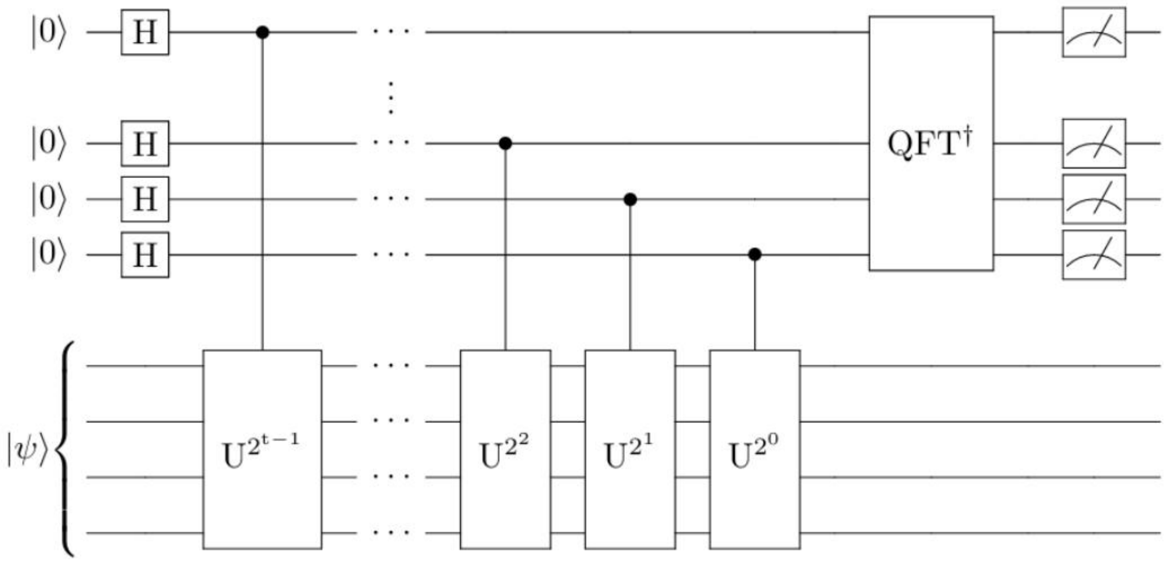


Figure 4: Circuit for Quantum Phase Estimation

Glasgow Paper, A quantum algorithm for gravitational wave matched filtering (Gao S. et al. (2021))

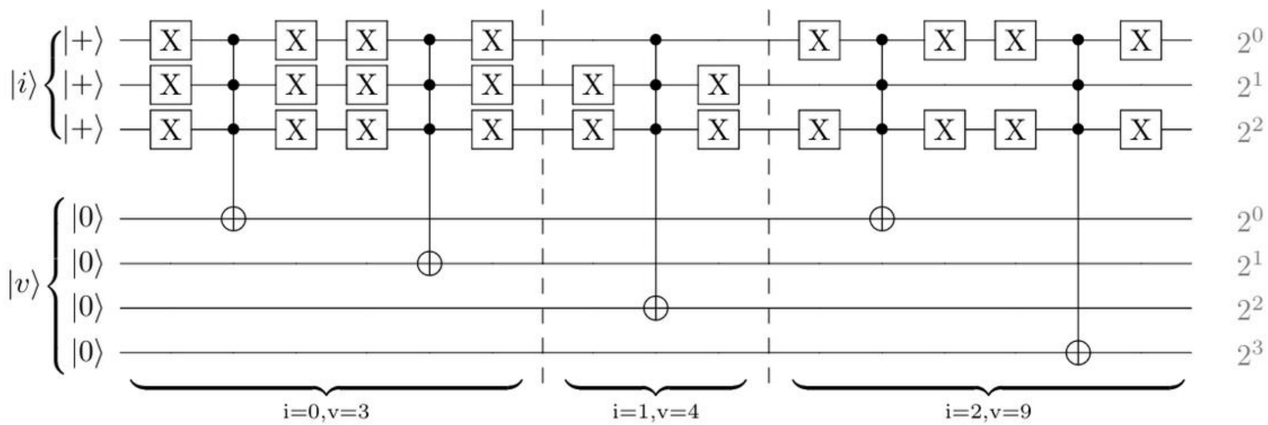
Latest developments in Maastricht

(Almost) Quantum Matched Filtering

To create a working Matched Filtering circuit, we need to be able to

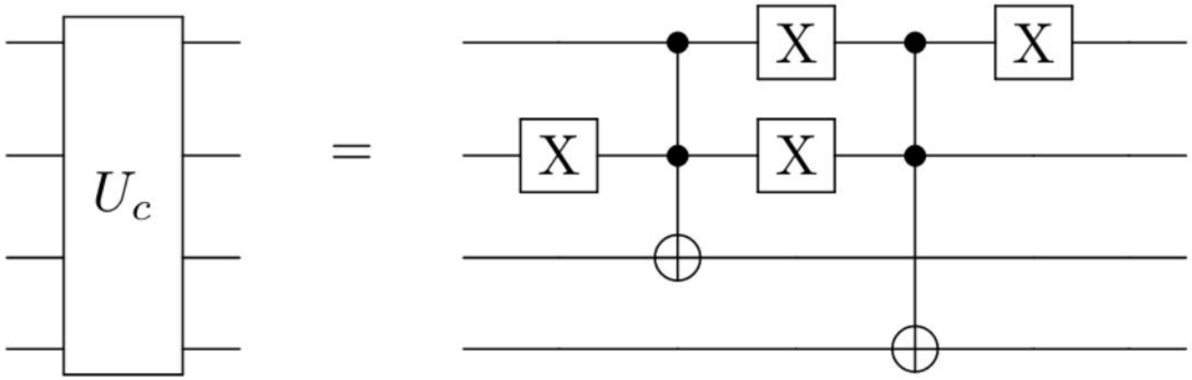
- Calculate the SNRs from time-series data (not yet solved)
- Encode these values into QC as an array
- Encode a certain threshold SNR, ρ_{thresh}
- Compare the values of the SNR database
- Return the indices of all the SNRs that crossed the threshold

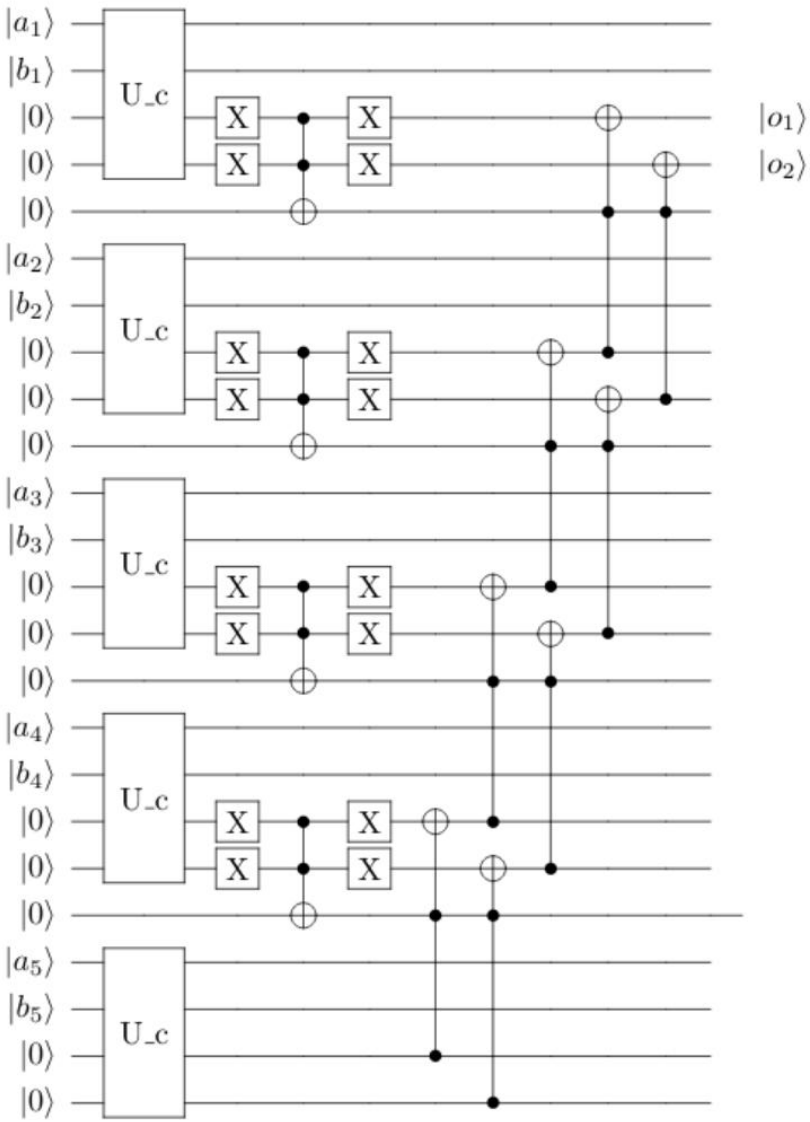
Database Encoding



Quantum Bit String Comparator

We need to be able to compare values of two bit strings. Luckily there already exists schematics for a Quantum Bit String Comparator (Oliveira et al. 2007).



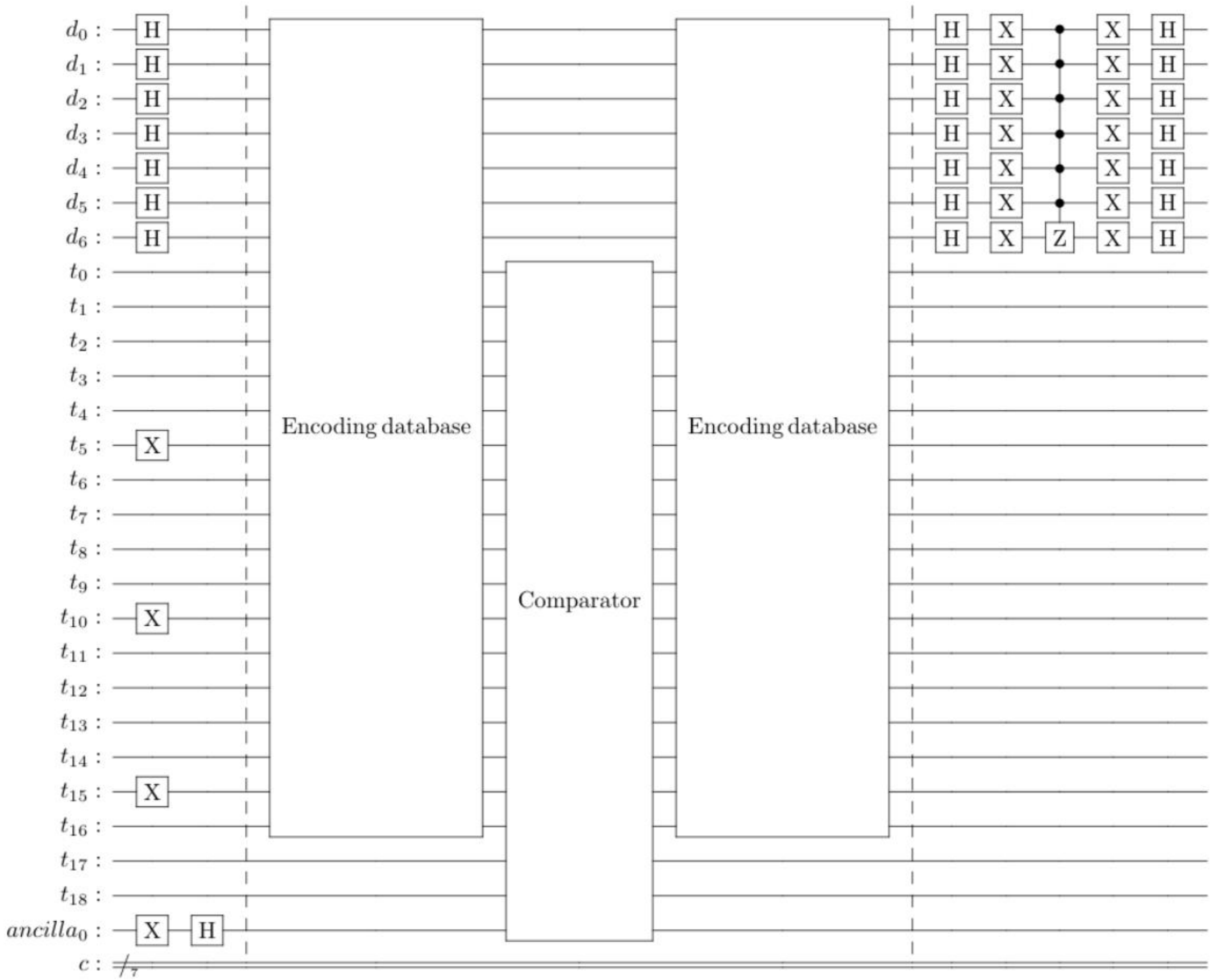


|a1⟩
|b1⟩
|0⟩
|0⟩
|0⟩
|a2⟩
|b2⟩
|0⟩
|0⟩
|0⟩
|a3⟩
|b3⟩
|0⟩
|0⟩
|0⟩
|a4⟩
|b4⟩
|0⟩
|0⟩
|0⟩
|a5⟩
|b5⟩
|0⟩
|0⟩
U_c
X
|o1⟩
|o2⟩

Creating the Grover Circuit

We need to combine:

- Data Encoding
- Bit String comparison
- Set ρ_{thresh}
- Disentangling
- Diffusion operator



d0 :
d1 :
d2 :
d3 :
d4 :
d5 :
d6 :
t0 :
t1 :
t2 :
t3 :
t4 :
t5 :
t6 :
t7 :
t8 :
t9 :
t10 :
t11 :
t12 :
t13 :
t14 :
t15 :
t16 :
t17 :
t18 :
ancilla0 :
c :
7
H
X
Z
Encoding database
Comparator
Encoding database

Capabilities

The circuit above was encoded using the values :

$$[3, 8, 1, 9, 0, 8, 0, 5, 7, 5, 3, 2, 5, 7, 5, 3, 2, 6, 8, 5, 6, 4, 3, 7, 5, 4, 3, 5, 6, 7, 4, 3, 4, 6, 7$$
$$8, 5, 3, 2, 2, 3, 4, 5, 4, 3, 2, 3, 4, 5, 6, 4, 3, 3, 3, 4, 5, 2, 2, 3, 4, 4, 4, 3, 2, 3, 4, 5, 6, 7, 6, 5]$$

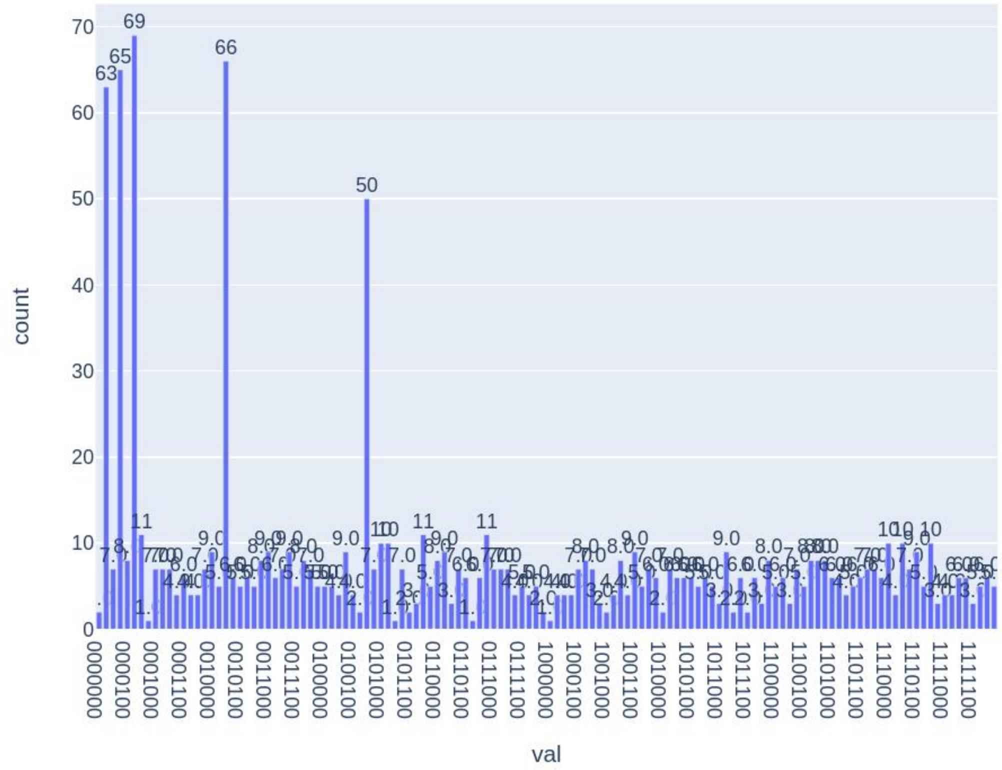
and

$$\rho_{thresh} = 7$$

This means we should be able to get an output of indices:

- 1
- 3
- 5
- 18
- 38

```
Total time:  48.63046964100795
```




```
Index 1 holds the value 8, which is above the chosen threshold.
Index 3 holds the value 9, which is above the chosen threshold.
Index 5 holds the value 8, which is above the chosen threshold.
Index 18 holds the value 8, which is above the chosen threshold.
```

Breaking down huge databases into smaller parts

At the moment, not enough qbits or processing power to run a full database. But by splicing:

- makes it possible to run on QC simulation (we have managed to run array length of 2^{17})
- lowers the hardware requirements for initial testing

Other possible benefit (currently investigated):

- Quantum counting has polynomial operational complexity
- Grover's $O(\sqrt{N})$

That means that we can possibly find an optimal splicing of parameter space to minimize some type of complexity

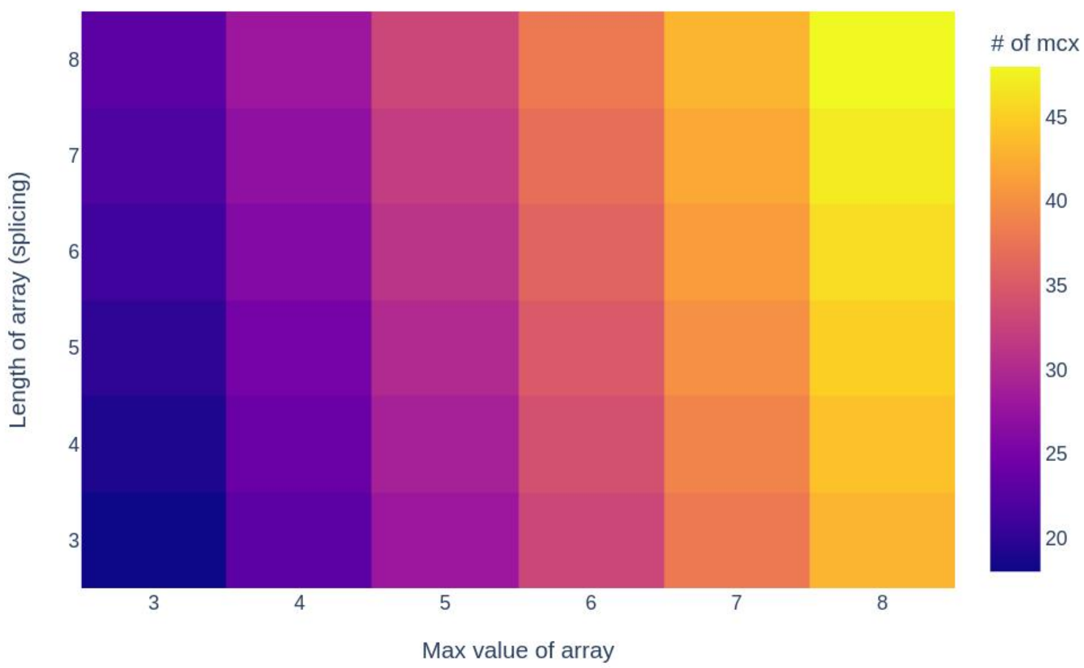
Testing for complexity

Measurment of Complexity

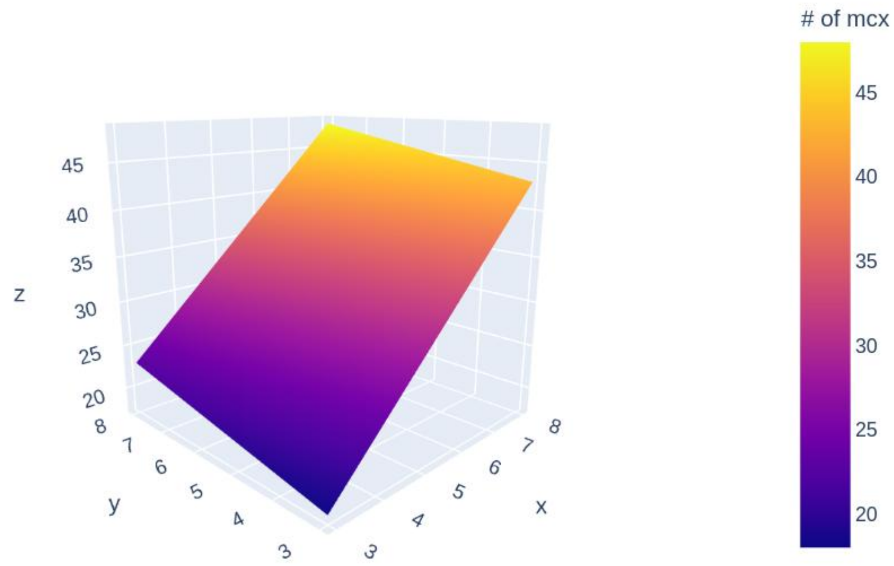
We have found it difficult to come up with a satisfacory measure:

- Cannot really test time complexity on a simulator. No clear idea on speed of individual gates.
- We can however measure the evolution of the number of gates depending on the input

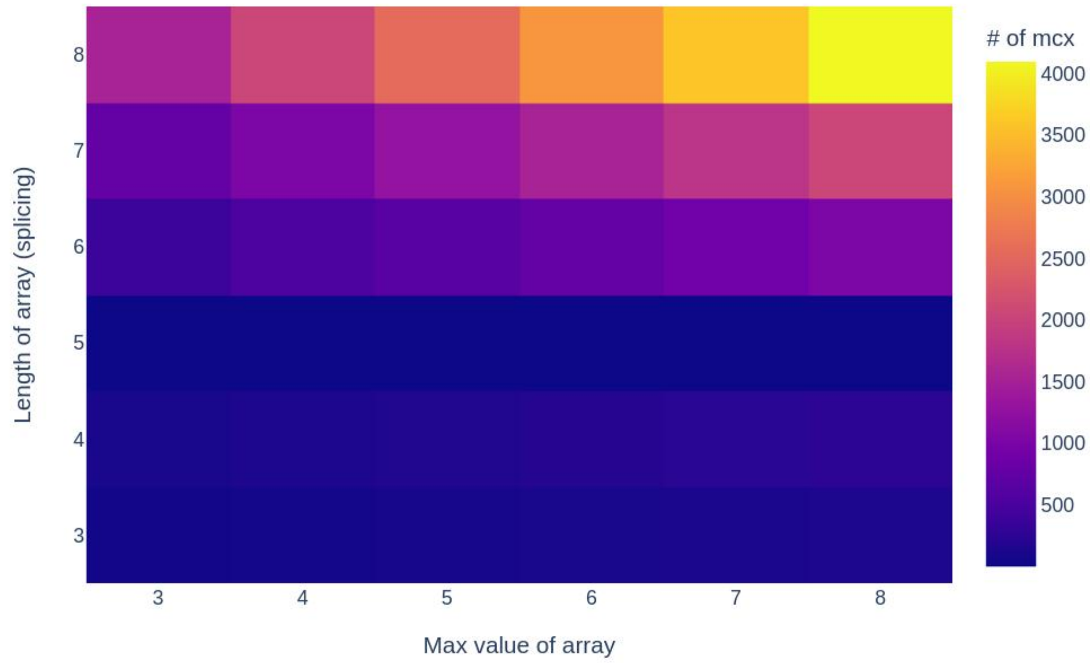
Impact of database and value size on Qbit number



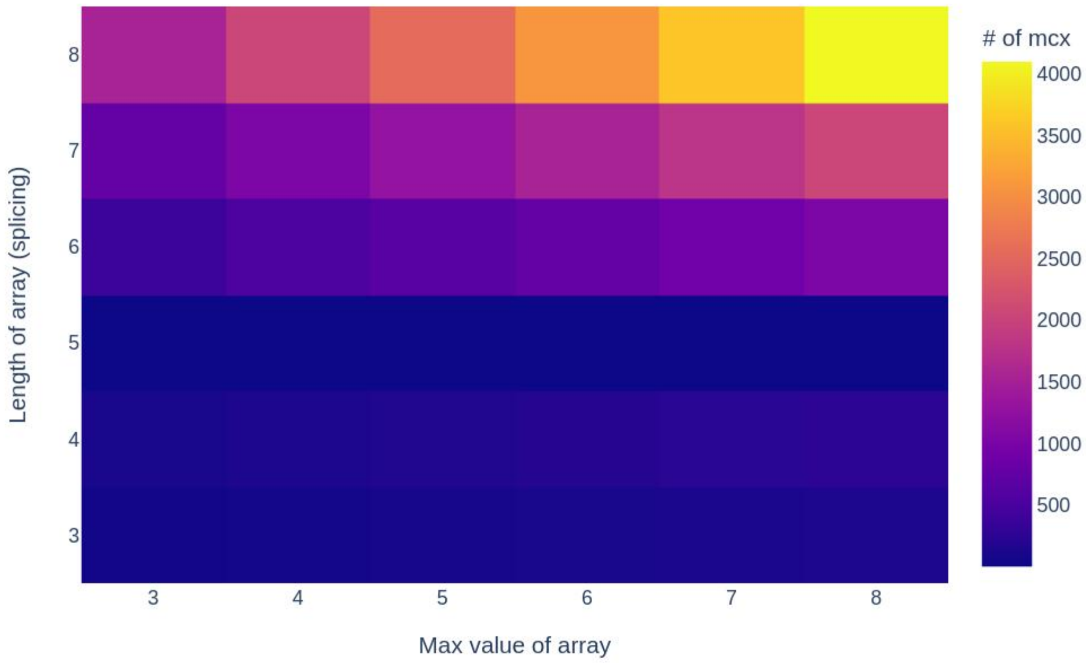
Impact of database and value size on Qbit number

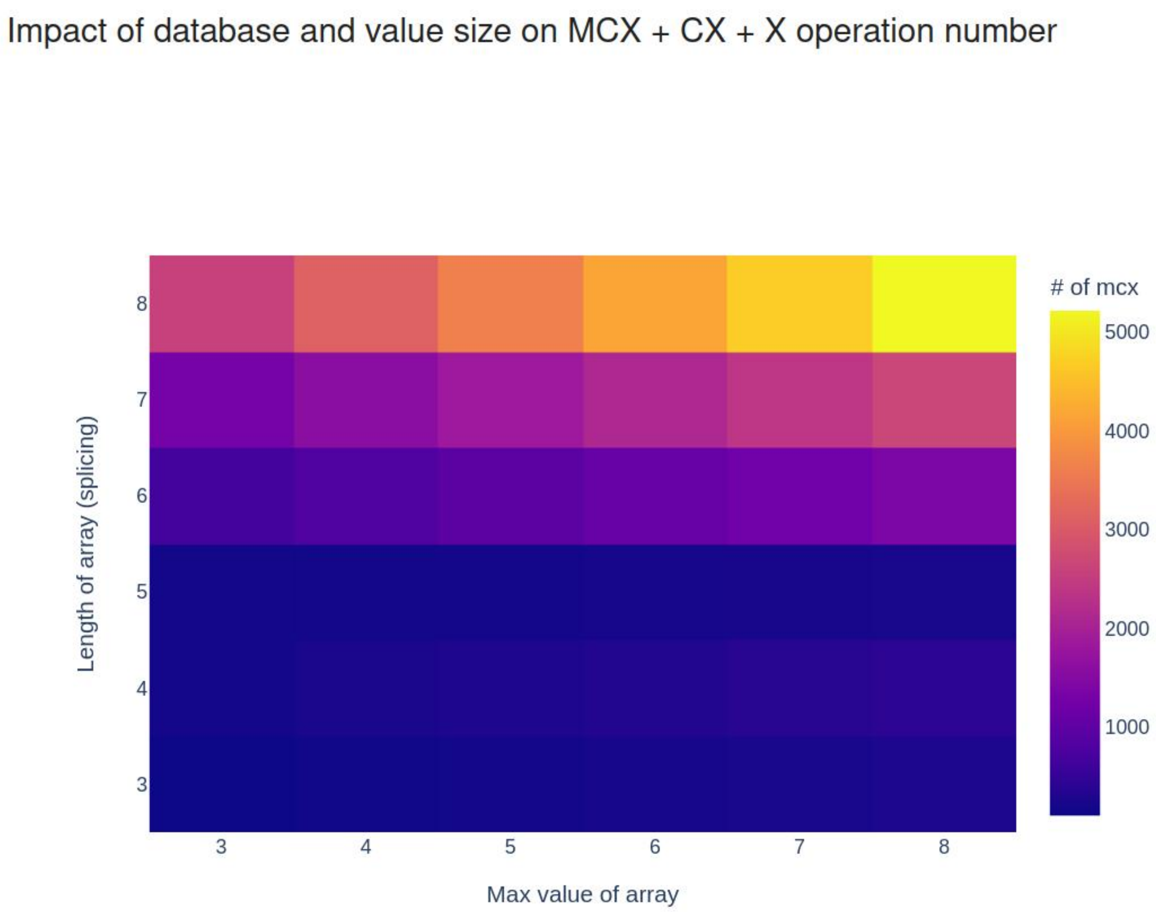


Impact of database and value size on MCX operation number



Impact of database and value size on MCX + CX operation number




Impact of database and value size on MCX + CX + X operation number
# of mcx
5000
4000
3000
2000
1000
Length of array (splicing)
8
7
6
5
4
3
Max value of array
3
4
5
6
7
8