Bachelor Project

Asmus Tørsleff

April 2023

Contents

1 Motivation **3**

2 Dense Simulation **4**
2.1 Representing a one qubit state 4
2.2 Multiple qubit states 5
2.3 Manipulating qubits 7
2.4 Manipulating qubits individually 9
2.5 Controlled gates 11
2.6 The Quantum Fourier Transform 12
2.7 Putting it all together 14

3 Tensor Network Simulation **16**
3.1 Contracting tensors 17
3.2 Reshaping tensors 17
3.3 Splitting two qubit gates into two tensors with a shared index . 17
3.4 Repressenting a circuit and state using tensors 19

4 Results & conclusion **24**

5 Further work & acknowledgements **27**

6 Appendix **29**
6.1 calculations for section 2.3 29
6.2 calculations for section 2.4 31

1 Motivation

The Fourier transfrom is one of the most widely applicable procedures, used in anything from signal processing and AI to seismology. Therefor it is of interest to explore different ways of computing it.

A variant of it; the discrete Fourier transfrom takes a number of equally spaced points on a function and transforms them into shifts and frequencies for a liniear combination of phase shifted sinus functions of different frequencies. We can also do the opposite.

We can use this e.g. to remove an annoying high pitch noice in a recording: We sample the sound wave, do the Fourier transformation to get the frequencies involved, take out the higher pitced ones and do the inverse Fourier transfrom to recover the sound wave, now without the high pitched noice.

There are many ways of computing this discrete Fourier transform, one of the more well known approaches is the fast Fourier transform (FFT), which has been our most efficient way of computing it for a while. There is an analogus algorithm designed for quantum computers namely the quantum Fourier transform (QFT).

In addition to its standalone value it is also widely used as a component of other quantum algorithms such as Shor's algorithm, and is therefor an intersting subject of improvement. The QFT has a significantly better time complexity, but only when run on a quantum computer. The QFT takes $O(n^2)$ quantum gates and FFT takes $O(log_2(2^n)2^n) = O(n2^n)$[2] classical logic gates to compute for 2^n amplitudes or data points. For the QFT n corresponds to the required number of qubits.

Recently there has been advancements in simulating the QFT approximatly on classical computers, while retaining some of this speedup, which is what this project focuses on. This has been shown in the paper[1] which will be refered to as 'the paper' in this paper. We will go though the math needed to implement two different simulations of the QFT, one exact and one approximate but vastly faster and more memmory efficient. The first only uses basic matrix math, the second requires a basic understanding of tensor networks which we will try to provide. This project also includes code implementing all the math touched upon by this paper and is intended as a learning resource, this means that the formulas shown will often not be the pretty driviations seen elsewhere but will seek to represent the individual bits that went into constructing them.

In the section 2 we will show how to use matrix math to simulate the QFT, and in section 3 we will show how to compute an exponentially close approximation using tensors. We will then compare the time taken, the memory used and the approximation error between the approaches in section 4. In the appendix there will be example calculations for many of the eqations shown later in this paper.

2 Dense Simulation

We want to simulate applying the QFT to some state, for this we need a way to represent the QFT and the state. In this section we will work towards representing the QFT circuit as a matrix and the state as a vector.

2.1 Representing a one qubit state

In classical computation a bit can either be 1 or 0, in quantum systems however a qubit has a certain probability of collapsing to either one or zero at a given time. To simulate this we use a vector to represent a qubit or a system of qubits, we denote this using Dirac notation. In Dirac notation we have a bra $\langle|$ and a ket $|\rangle$, generally a bra denotes a row vector and a ket denotes a column vector. To extract the probabilities of a qubit or system of qubits collapsing to one state or the other we take the norm squared of the entries in the vector, the sum of these must be 1 to be a valid state.

$$|\mathbf{1}\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Figure 1: A qubit with a probability of 1 of collapsing to one

$$|\mathbf{0}\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

Figure 2: A qubit with a probability of 1 of collapsing to zero

$$|+\rangle = \sqrt{0.5}(|\mathbf{0}\rangle + |\mathbf{1}\rangle) = \begin{bmatrix} \sqrt{0.5} \\ \sqrt{0.5} \end{bmatrix}$$
$$|-\rangle = \sqrt{0.5}(|\mathbf{0}\rangle - |\mathbf{1}\rangle) = \begin{bmatrix} \sqrt{0.5} \\ -\sqrt{0.5} \end{bmatrix}$$

Figure 3: Qubits with a probability of 0.5 of collapsing to zero and one

In general a qubit $\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \alpha\,|\mathbf{0}\rangle + \beta\,|\mathbf{1}\rangle$ has a probability of $|\alpha|^2$ of collapsing to zero and $|\beta|^2$ of collapsing to one. Since we use a vector of complex numbers $|x|^2 = xx^*$ denotes the norm squared, where if $x = a + bi$ then x^* is the complex

conjugate of x, $x^* = a - bi$. For readabilitys sake we will not write $+bi$ after a number if $b = 0$, you can assume all numbers in vectors and matricies are complex.

If we look at these vectors in a coordinate system, where the z axis is the imaginary axis, they all lie on a unit circle.

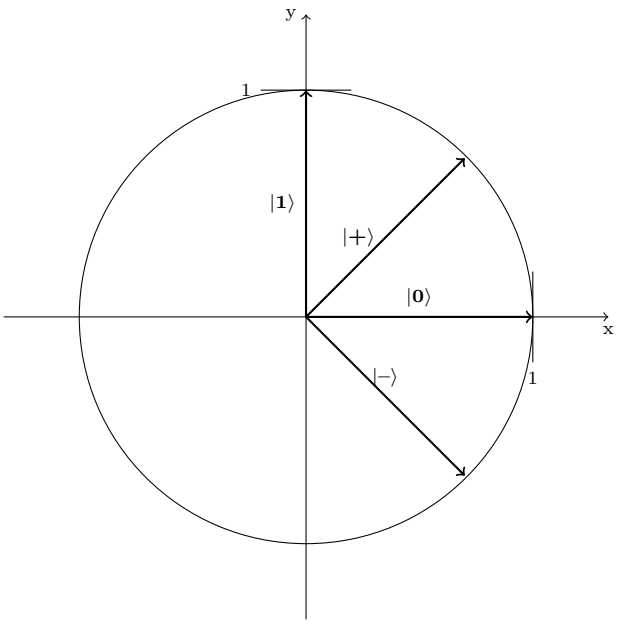

Figure 4: 'Bloch circle' with vectors $|\mathbf{1}\rangle$, $|+\rangle$, $|\mathbf{0}\rangle$ and $|-\rangle$ plotted

In reality we often use complex numbers with a non-zero imaginary part in the vectors, which results in the qubit states instead lying on the unit sphere, this is known as the Bloch sphere. The z axis is called the phase and generally has no impact on the probabilities of collapsing to a certain state, however it can be used to hold extra information before the qubit is collapsed, this can be useful in computations.

2.2 Multiple qubit states

$|\mathbf{1}+-\rangle$ denotes a system collapsing to either $|\mathbf{100}\rangle$, $|\mathbf{101}\rangle$, $|\mathbf{110}\rangle$ or $|\mathbf{111}\rangle$ with probability 0.25. If we want to compute the state vector for the whole system we take the Kronecker product, sometimes called the tensor product, of the vectors representing the individual qubits in the system. The Kronecker product of two matrix like objects consists of duplicating the second object for every entry in the first object and then scaling every value in the duplicate by the entry in the first object.

$$\begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \end{bmatrix} \otimes \begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \\ b_{20} & b_{21} \end{bmatrix} =$$

$$\begin{bmatrix} a_{00}\begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \\ b_{20} & b_{21} \end{bmatrix} & a_{01}\begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \\ b_{20} & b_{21} \end{bmatrix} & a_{02}\begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \\ b_{20} & b_{21} \end{bmatrix} \\ a_{10}\begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \\ b_{20} & b_{21} \end{bmatrix} & a_{11}\begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \\ b_{20} & b_{21} \end{bmatrix} & a_{12}\begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \\ b_{20} & b_{21} \end{bmatrix} \end{bmatrix} =$$

$$\begin{bmatrix} a_{00}b_{00} & a_{00}b_{01} & a_{01}b_{00} & a_{01}b_{01} & a_{02}b_{00} & a_{02}b_{01} \\ a_{00}b_{10} & a_{00}b_{11} & a_{01}b_{10} & a_{01}b_{11} & a_{02}b_{10} & a_{02}b_{11} \\ a_{00}b_{20} & a_{00}b_{21} & a_{01}b_{20} & a_{01}b_{21} & a_{02}b_{20} & a_{02}b_{21} \\ a_{10}b_{00} & a_{10}b_{01} & a_{11}b_{00} & a_{11}b_{01} & a_{12}b_{00} & a_{12}b_{01} \\ a_{10}b_{10} & a_{10}b_{11} & a_{11}b_{10} & a_{11}b_{11} & a_{12}b_{10} & a_{12}b_{11} \\ a_{10}b_{20} & a_{10}b_{21} & a_{11}b_{20} & a_{11}b_{21} & a_{12}b_{20} & a_{12}b_{21} \end{bmatrix}$$

Figure 5: Example of taking the Kronecker product of two matricies

$$|\mathbf{1}+-\rangle = |\mathbf{1}\rangle \otimes |+\rangle \otimes |-\rangle =$$

$$\begin{bmatrix} 0 \\ 1 \end{bmatrix} \otimes \begin{bmatrix} \sqrt{0.5} \\ \sqrt{0.5} \end{bmatrix} \otimes \begin{bmatrix} \sqrt{0.5} \\ -\sqrt{0.5} \end{bmatrix} =$$

$$\begin{bmatrix} 0\begin{bmatrix} \sqrt{0.5}\begin{bmatrix} \sqrt{0.5} \\ -\sqrt{0.5} \end{bmatrix} \\ \sqrt{0.5}\begin{bmatrix} \sqrt{0.5} \\ -\sqrt{0.5} \end{bmatrix} \end{bmatrix} \\ 1\begin{bmatrix} \sqrt{0.5}\begin{bmatrix} \sqrt{0.5} \\ -\sqrt{0.5} \end{bmatrix} \\ \sqrt{0.5}\begin{bmatrix} \sqrt{0.5} \\ -\sqrt{0.5} \end{bmatrix} \end{bmatrix} \end{bmatrix} = \begin{bmatrix} 0\begin{bmatrix} 0.5 \\ -0.5 \\ 0.5 \\ -0.5 \end{bmatrix} \\ 1\begin{bmatrix} 0.5 \\ -0.5 \\ 0.5 \\ -0.5 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0.5 \\ -0.5 \\ 0.5 \\ -0.5 \end{bmatrix} =$$

$$0 \cdot |\mathbf{000}\rangle + 0 \cdot |\mathbf{001}\rangle + 0 \cdot |\mathbf{010}\rangle + 0 \cdot |\mathbf{011}\rangle +$$
$$0.5 \cdot |\mathbf{100}\rangle - 0.5 \cdot |\mathbf{101}\rangle + 0.5 \cdot |\mathbf{110}\rangle - 0.5 \cdot |\mathbf{111}\rangle$$

Figure 6: Using the Kronecker product to compute the state vector for the system in state $|\mathbf{1}+-\rangle$

If we square this vector, equvilent to takeing the norm squared when the imaginary parts are zero, we get the probability corresponding to each state.

$$|\,|\mathbf{1}+-\rangle\,|^2 = \begin{bmatrix} 0^2 \\ 0^2 \\ 0^2 \\ 0^2 \\ 0.5^2 \\ -0.5^2 \\ 0.5^2 \\ -0.5^2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{bmatrix}$$

Figure 7: Squaring entries the in state vector to obtain probabilities of the system being in each state, the i'th entry in the vector is the probability of the system being in state $|i\rangle$ if you write i in binary using 3 digits

2.3 Manipulating qubits

In traditional electrical circuits we use logic gates such as NOT, AND, OR, NAND, etc. to manipulate bits. In quantum circuits some common gates are Pauli-X (X, bit flip or NOT), I (identity), H (Hadamard), CNOT (CX or controlled not), etc. In the same way we represent the state of a quantum system as a vector, we can represent a gate that operates on a state as a matrix.

$$\mathbf{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \mathbf{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad \mathbf{H} = \begin{bmatrix} \sqrt{0.5} & \sqrt{0.5} \\ \sqrt{0.5} & -\sqrt{0.5} \end{bmatrix} \quad \mathbf{R_n} = \begin{bmatrix} 1 & 0 \\ 0 & \mathrm{e}^{\frac{2\pi \mathrm{i}}{2^n}} \end{bmatrix}$$

Figure 8: Matrixes corresponding to $\mathbf{I}$, $\mathbf{X}$, $\mathbf{H}$ and $\mathbf{R_n}$ gates, which act on a single cubit

If we wish to apply a gate to a qubit in a system we do so by doing matrix multiplication between the state and the gate.

$$\mathbf{I}\,|\mathbf{0}\rangle = |\mathbf{0}\rangle \quad \mathbf{I}\,|\mathbf{1}\rangle = |\mathbf{1}\rangle$$

Figure 9: Applying $\mathbf{I}$ gate to the states $|\mathbf{0}\rangle$ and $|\mathbf{1}\rangle$

$$\mathbf{X}\,|\mathbf{0}\rangle = |\mathbf{1}\rangle \quad \mathbf{X}\,|\mathbf{1}\rangle = |\mathbf{0}\rangle$$

Figure 10: Applying $\mathbf{X}$ gate to the states $|\mathbf{0}\rangle$ and $|\mathbf{1}\rangle$

$$\mathbf{H}\,|\mathbf{0}\rangle = |+\rangle \quad \mathbf{H}\,|\mathbf{1}\rangle = |-\rangle$$

Figure 11: Applying $\mathbf{H}$ gate to the states $|\mathbf{0}\rangle$ and $|\mathbf{1}\rangle$

$$\mathbf{R_2}\,|\mathbf{0}\rangle = |\mathbf{0}\rangle \quad \mathbf{R_2}\,|\mathbf{1}\rangle = \begin{bmatrix} 0 \\ \mathrm{i} \end{bmatrix}$$

Figure 12: Applying $\mathbf{R_2}$ gate to the states $|\mathbf{0}\rangle$ and $|\mathbf{1}\rangle$

$$\mathbf{CX}\,|\mathbf{00}\rangle = |\mathbf{00}\rangle \quad \mathbf{CX}\,|\mathbf{01}\rangle = |\mathbf{01}\rangle \quad \mathbf{CX}\,|\mathbf{10}\rangle = |\mathbf{11}\rangle \quad \mathbf{CX}\,|\mathbf{11}\rangle = |\mathbf{10}\rangle$$

Figure 13: Applying $\mathbf{CX}$ gate to the states $|\mathbf{00}\rangle$, $|\mathbf{01}\rangle$, $|\mathbf{10}\rangle$ and $|\mathbf{11}\rangle$

See appendix 6.1 for detailed calculations of these applications.

Here we see that applying $\mathbf{I}$ does not change the state. Applying $\mathbf{X}$ takes the state to the same state but reversed. Applying $\mathbf{H}$ takes the state into a super position of $|\mathbf{1}\rangle$ and $|\mathbf{0}\rangle$. Applying $\mathbf{R_n}$ changes the phase of the cubit but not the chance of it collapsing to something. A key property of quantum gates is that they are reversible, and in fact their own inverses.

$$\mathbf{I}(\mathbf{I}\,|\mathbf{0}\rangle) = |\mathbf{0}\rangle \quad \mathbf{I}(\mathbf{I}\,|\mathbf{1}\rangle) = |\mathbf{1}\rangle$$

Figure 14: Applying $\mathbf{I}$ gate to the states $|\mathbf{0}\rangle$ and $|\mathbf{1}\rangle$ twice

$$\mathbf{X}(\mathbf{X}\,|\mathbf{0}\rangle) = |\mathbf{0}\rangle \quad \mathbf{X}(\mathbf{X}\,|\mathbf{1}\rangle) = |\mathbf{1}\rangle$$

Figure 15: Applying $\mathbf{X}$ gate to the states $|\mathbf{0}\rangle$ and $|\mathbf{1}\rangle$ twice

$$\mathbf{H}(\mathbf{H}\,|\mathbf{0}\rangle) = |\mathbf{0}\rangle \quad \mathbf{H}(\mathbf{H}\,|\mathbf{1}\rangle) = |\mathbf{1}\rangle$$

Figure 16: Applying $\mathbf{H}$ gate to the states $|\mathbf{0}\rangle$ and $|\mathbf{1}\rangle$ twice

$$\mathbf{R_2}(\mathbf{R_2}\,|\mathbf{0}\rangle) = |\mathbf{0}\rangle \quad \mathbf{R_2}(\mathbf{R_2}\,|\mathbf{1}\rangle) = |\mathbf{1}\rangle$$

Figure 17: Applying $\mathbf{R_2}$ gate to the states $|\mathbf{0}\rangle$ and $|\mathbf{1}\rangle$ twice

$$\mathbf{CX}(\mathbf{CX}\,|\mathbf{00}\rangle) = |\mathbf{00}\rangle \quad \mathbf{CX}(\mathbf{CX}\,|\mathbf{01}\rangle) = |\mathbf{01}\rangle \quad \mathbf{CX}(\mathbf{CX}\,|\mathbf{10}\rangle) = |\mathbf{10}\rangle \quad \mathbf{CX}(\mathbf{CX}\,|\mathbf{11}\rangle) = |\mathbf{11}\rangle$$

Figure 18: Applying $\mathbf{CX}$ gate to the states $|\mathbf{00}\rangle$, $|\mathbf{01}\rangle$, $|\mathbf{10}\rangle$ and $|\mathbf{11}\rangle$ twice

This makes any quantum circuit constructed using these gates reversible.

2.4 Manipulating qubits individually

We have seen how we apply a gate to a system of qubits the same size as the gate. If we want to apply a gate to the i'th qubit we construct a gate the same size as the system which acts with identity on every other qubit. We again use the Kronecker product for this.

$$(\mathbf{I} \otimes \mathbf{I} \otimes \mathbf{X})\,|\mathbf{000}\rangle = |\mathbf{001}\rangle$$

Figure 19: Applying $\mathbf{X}$ to the first qubit in a 3 qubit system

$$(\mathbf{I} \otimes \mathbf{X} \otimes \mathbf{I})\,|\mathbf{000}\rangle = |\mathbf{010}\rangle$$

Figure 20: Applying $\mathbf{X}$ to the second qubit in a 3 qubit system

$$(\mathbf{X} \otimes \mathbf{I} \otimes \mathbf{I})\,|\mathbf{000}\rangle = |\mathbf{100}\rangle$$

Figure 21: Applying $\mathbf{X}$ to the thrid qubit in a 3 qubit system

If we want to apply multiple gates at a time we just replace the corresponding $\mathbf{I}$ gates.

$$(\mathbf{X} \otimes \mathbf{I} \otimes \mathbf{H}) |\mathbf{000}\rangle = |\mathbf{10+}\rangle$$

Figure 22: Applying **X** to the thrid qubit and **H** to the first qubit in a 3 qubit system

See calculations in appendix 6.2.

In general we can construct and apply any number of gates, $U_0, U_1, .., U_n$, to a state $|\Psi\rangle$ in sequence, this is in essence our quantum circuit. And because of the assosiative nature of matrix multiplication we can combine the entire circuit before applying it to different states! This saves tremendous amounts of resources when doing repetitative computations.

$$U_0 U_1 .. U_n |\Psi\rangle = \left(\prod_{i=0}^{n} U_i \right) |\Psi\rangle$$

Figure 23: Associative nature of matrix multiplication

$$\mathbf{U} = \mathbf{CX} (\mathbf{I} \otimes \mathbf{H}) (\mathbf{I} \otimes \mathbf{X}) = \begin{bmatrix} \sqrt{0.5} & \sqrt{0.5} & 0 & 0 \\ -\sqrt{0.5} & \sqrt{0.5} & 0 & 0 \\ 0 & 0 & -\sqrt{0.5} & \sqrt{0.5} \\ 0 & 0 & \sqrt{0.5} & \sqrt{0.5} \end{bmatrix}$$

Figure 24: Computing a matrix representation of a circuit,**U**, that first applies **X** then **H** to the first qubit then **CX** to the first and second qubit

$$\begin{aligned} \mathbf{U} |\mathbf{00}\rangle &= |\mathbf{0-}\rangle \\ \mathbf{U} |\mathbf{01}\rangle &= |\mathbf{0+}\rangle \\ \mathbf{U} |\mathbf{10}\rangle &= -|\mathbf{1-}\rangle \\ \mathbf{U} |\mathbf{11}\rangle &= |\mathbf{1+}\rangle \end{aligned}$$

Figure 25: Applying the circuit represented by the matrix **U** from figure 24 to different states

2.5 Controlled gates

We have described the $\mathbf{R_n}$ gate but for the QFT circuit we need it controlled by another qubit. We can construct a controlled gate out of any other gate, we simply take the gate that should happen if the control qubit is $|\mathbf{0}\rangle$ and place it in the top left qudrant of a matrix, then place what should happen if it is $|\mathbf{1}\rangle$ in the bottom right qudrant. This can be expressed like so:

$$\begin{gathered}\mathbf{CU_{1,2}} = |\mathbf{0}\rangle\langle\mathbf{0}| \otimes \mathbf{I} + |\mathbf{1}\rangle\langle\mathbf{1}| \otimes \mathbf{U} = \\ \begin{bmatrix}1\\0\end{bmatrix}\begin{bmatrix}1 & 0\end{bmatrix} \otimes \mathbf{I} + \begin{bmatrix}0\\1\end{bmatrix}\begin{bmatrix}0 & 1\end{bmatrix} \otimes \mathbf{U} = \\ \begin{bmatrix}1 & 0\\0 & 0\end{bmatrix} \otimes \mathbf{I} + \begin{bmatrix}0 & 0\\0 & 1\end{bmatrix} \otimes \mathbf{U}\end{gathered}$$

Figure 26: Constructing a controled gate $\mathbf{CU_{1,2}}$ from a gate $\mathbf{U}$ where the first cubit controlls the second

$$\begin{gathered}\mathbf{CU_{2,1}} = \mathbf{I} \otimes |\mathbf{0}\rangle\langle\mathbf{0}| + \mathbf{U} \otimes |\mathbf{1}\rangle\langle\mathbf{1}| = \\ \mathbf{I} \otimes \begin{bmatrix}1\\0\end{bmatrix}\begin{bmatrix}1 & 0\end{bmatrix} + \mathbf{U} \otimes \begin{bmatrix}0\\1\end{bmatrix}\begin{bmatrix}0 & 1\end{bmatrix} = \\ \mathbf{I} \otimes \begin{bmatrix}1 & 0\\0 & 0\end{bmatrix} + \mathbf{U} \otimes \begin{bmatrix}0 & 0\\0 & 1\end{bmatrix}\end{gathered}$$

Figure 27: Constructing a controled gate $\mathbf{CU_{2,1}}$ from a gate $\mathbf{U}$ where the second cubit controlls the first

In our case the controlling qubit is always after the $\mathbf{R_n}$ gate so we only need the second formula. It is important to note that the controlled gate, $\mathbf{U}$ in the examples, can be of any size, therefor it is not an issue that some of the controlled gates have 'empty' wires between them for example here is the result for a $\mathbf{CR_{3_{3,1}}}$ gate.

$$\mathbf{CR_{3_{3,1}}} = \mathbf{I} \otimes \mathbf{I} \otimes |\mathbf{0}\rangle \langle \mathbf{0}| + \mathbf{R_3} \otimes \mathbf{I} \otimes |\mathbf{1}\rangle \langle \mathbf{1}| =$$
$$\mathbf{I} \otimes \mathbf{I} \otimes \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix} + \mathbf{R_3} \otimes \mathbf{I} \otimes \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \end{bmatrix} =$$
$$\mathbf{I} \otimes \mathbf{I} \otimes \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + \mathbf{R_3} \otimes \mathbf{I} \otimes \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$$

Figure 28: Constructing a controled gate $\mathbf{CR_{3_{3,1}}}$ from a gate $\mathbf{R_3}$ where the second cubit controlls the first

$|\Psi\rangle \langle \Psi|$ is often called a projection on Ψ.

2.6 The Quantum Fourier Transform

Now we can represent a state as a vector and we have the building blocks for representing the QFT as a matrix. The QFT_n circuit applies the QFT to a n qubit state and looks like this:

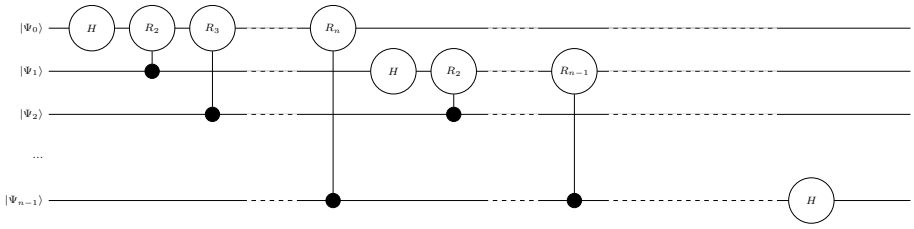

Figure 29: circuit diagram of QFT_n

In essance this circuit diagram can be read:

1. apply the $\mathbf{H}$ gate to the first qubit.
2. apply the $\mathbf{R_2}$ gate to the first qubit controlled by the second qubit.
3. apply the $\mathbf{R_3}$ gate to the first qubit controlled by the third qubit.
4. ...
5. apply the $\mathbf{R_n}$ gate to the first qubit controlled by the last qubit.
6. apply the $\mathbf{H}$ gate to the second qubit.
7. apply the $\mathbf{R_2}$ gate to the second qubit controlled by the third qubit.
8. ...
9. apply the $\mathbf{R_{n-1}}$ gate to the second qubit controlled by the last qubit.

10. ...

11. apply the **H** gate to the last qubit.

This is all well and good, but how does this circuit end up performing a discreet Fourier transform? The definition of the DFT is given by this formula:

$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-\frac{i2\pi}{N}kn}$$

where x is the input vector and X is the output vector, the subscripts are used for indexing in the vectors.

$$x_n = \frac{1}{N}\sum_{k=0}^{N-1} X_k \cdot e^{\frac{i2\pi}{N}kn}$$

This is the inverse DFT. The QFT acts on a state $|x\rangle = |x_0, x_1, ..., x_{N-1}\rangle$ and transforms it into the state $|y\rangle = |y_0, y_1, ..., y_{N-1}\rangle$, the transformation is exactly:

$$y_k = \frac{1}{\sqrt{N}}\sum_{n=0}^{N-1} x_n e^{\frac{i2\pi}{N}kn}$$

We see that it is eqiuvalent to the inverse DFT. The QFT applied to a state is also sometimes written as

$$\mathrm{QFT}\,|x_1x_2\ldots x_n\rangle = \tfrac{1}{\sqrt{N}}\ (|\mathbf{0}\rangle + e^{2\pi i\,[0.x_n]}\,|\mathbf{1}\rangle) \otimes (|\mathbf{0}\rangle + e^{2\pi i\,[0.x_{n-1}x_n]}\,|\mathbf{1}\rangle) \otimes \cdots \otimes (|\mathbf{0}\rangle + e^{2\pi i\,[0.x_1x_2\ldots x_n]}\,|\mathbf{1}\rangle)$$

where $[b_0.b_1b_2b_3...b_n] = \Sigma_{i=0}^{n}\frac{b_i}{2^i}$, so a binary number[4]. If we don't apply the swaps in the end of the QFT we get

$$\mathrm{QFT}\,|x_1x_2\ldots x_n\rangle = \tfrac{1}{\sqrt{N}}\ (|\mathbf{0}\rangle + e^{2\pi i\,[0.x_1x_2\ldots x_n]}\,|\mathbf{1}\rangle) \otimes \cdots \otimes (|\mathbf{0}\rangle + e^{2\pi i\,[0.x_{n-1}x_n]}\,|\mathbf{1}\rangle) \otimes (|\mathbf{0}\rangle + e^{2\pi i\,[0.x_n]}\,|\mathbf{1}\rangle)$$

But how does the circuit laid out above result in this formula? Let us first look at $QFT_1\,|x_1\rangle$, which is just $\mathbf{H}\,|x_1\rangle$ this can be written as $\frac{1}{\sqrt{2^1}}(|\mathbf{0}\rangle + e^{\frac{2\pi i}{2^1}x_1}\,|\mathbf{1}\rangle)$. We see here that if x_1 is 1 we have $\frac{1}{\sqrt{2^1}}(|\mathbf{0}\rangle + e^{\pi i}\,|\mathbf{1}\rangle)$ and if it is 0 we have $\frac{1}{\sqrt{2^1}}(|\mathbf{0}\rangle + e^{0}\,|\mathbf{1}\rangle)$ whitch is equivalent to $\frac{1}{\sqrt{2^1}}(|\mathbf{0}\rangle - |\mathbf{1}\rangle) = H\,|\mathbf{1}\rangle$ and $\frac{1}{\sqrt{2^1}}(|\mathbf{0}\rangle + |\mathbf{1}\rangle) = H\,|\mathbf{0}\rangle$. To make the equation match the one from the definition we can say $\frac{1}{\sqrt{2^1}}(|\mathbf{0}\rangle + e^{\frac{2\pi i}{2^1}x_1}\,|\mathbf{1}\rangle) = \frac{1}{\sqrt{2^1}}(|\mathbf{0}\rangle + e^{2\pi i\frac{x_1}{2^1}}\,|\mathbf{1}\rangle) = \frac{1}{\sqrt{2^1}}(|\mathbf{0}\rangle + e^{2\pi i[0.x_1]}\,|\mathbf{1}\rangle)$.

For the case of $QFT_2\,|x_1, x_2\rangle$ we have that x_2 is transformed in the same way as x_1 in QFT_1 and x_1 is now transformed like so $\mathbf{H}\,|x_1\rangle$ if $x_2 = 0$ and $\mathbf{HR_2}\,|x_1\rangle$ if $x_2 = 1$. This can be expressed like so $\frac{1}{\sqrt{2^1}}(|\mathbf{0}\rangle + e^{\frac{2\pi i}{2^1}x_1}e^{\frac{2\pi i}{2^2}x_2}\,|\mathbf{1}\rangle) = \frac{1}{\sqrt{2^1}}(|\mathbf{0}\rangle + e^{\frac{2\pi i}{2^1}x_1 + \frac{2\pi i}{2^2}x_2}\,|\mathbf{1}\rangle) = \frac{1}{\sqrt{2^1}}(|\mathbf{0}\rangle + e^{2\pi i[0.x_1x_2]}\,|\mathbf{1}\rangle)$. If we put this together we get $\frac{1}{\sqrt{2^1}}(|\mathbf{0}\rangle + e^{2\pi i[0.x_1x_2]}\,|\mathbf{1}\rangle) \otimes \frac{1}{\sqrt{2^1}}(|\mathbf{0}\rangle + e^{2\pi i[0.x_2]}\,|\mathbf{1}\rangle) = \frac{1}{\sqrt{2^2}}(|\mathbf{0}\rangle + e^{2\pi i[0.x_1x_2]}\,|\mathbf{1}\rangle) \otimes (|\mathbf{0}\rangle + e^{2\pi i[0.x_2]}\,|\mathbf{1}\rangle)$.

For the case of $QFT_3 \left|x_1, x_2, x_3\right\rangle$ we have that x_2 and x_3 is transformed in the same way as x_1 and x_2 in QFT_2 and x_1 is now transformed like so $\mathbf{H}\left|x_1\right\rangle$ if $x_2 = x_3 = 0$, $\mathbf{HR_2}\left|x_1\right\rangle$ if $x_2 = 1, x_3 = 0$, $\mathbf{HR_3}\left|x_1\right\rangle$ if $x_2 = 0, x_3 = 1$ and $\mathbf{HR_2R_3}\left|x_1\right\rangle$ if $x_2 = x_3 = 1$. This can be expressed with $\frac{1}{\sqrt{2^1}}(\left|\mathbf{0}\right\rangle + e^{\frac{2\pi i}{2^1}x_1}e^{\frac{2\pi i}{2^2}x_2}e^{\frac{2\pi i}{2^3}x_3}\left|\mathbf{1}\right\rangle) = \frac{1}{\sqrt{2^1}}(\left|\mathbf{0}\right\rangle + e^{\frac{2\pi i}{2^1}x_1+\frac{2\pi i}{2^2}x_2+\frac{2\pi i}{2^3}x_3}\left|\mathbf{1}\right\rangle) = \frac{1}{\sqrt{2^1}}(\left|\mathbf{0}\right\rangle + e^{2\pi i[0.x_1x_2x_3]}\left|\mathbf{1}\right\rangle)$.

This pattern continues in a sort of recurcive definition of QFT_n in terms of QFT_{n-1} and one 'step' affecting the first qubit.

2.7 Putting it all together

With these tools we can now create a formula for constructing a n qubit state and a n qubit QFT circuit.

$$\left|\Psi\right\rangle = \left|\Psi_1\Psi_2..\Psi_n\right\rangle = \bigotimes_{i=1}^{n}\left|\Psi_i\right\rangle$$

Figure 30: Constructing the state vector for the n qubit state $\left|\Psi\right\rangle$

$$I(k) = \begin{cases} \bigotimes_{i=k}^{1}\mathbf{I} & \text{if } k \geq 1 \\ \left[1\right] & \text{otherwise} \end{cases}$$

$$STEP(i,n) = \begin{cases} \prod_{j=i}^{2}\left(I(i)\otimes(\mathbf{I}\otimes I(j-2)\otimes\left|\mathbf{0}\right\rangle\left\langle\mathbf{0}\right| + \mathbf{R_j}\otimes I(j-2)\otimes\left|\mathbf{1}\right\rangle\left\langle\mathbf{1}\right|)\otimes I(n-i-j)\right) & \text{if } i \geq 2 \\ \left[1\right] & \text{otherwise} \end{cases}$$

$$QFT(n) = \prod_{i=n-1}^{0}\left(STEP(n-i,n)\left(I(i)\otimes\mathbf{H}\otimes I(n-i-1)\right)\right)$$

Figure 31: Constructing the matrix for the n qubit QFT

This can be turned into pseudo code:

```
STATE(psi)
1  r = [1]
2  for each digit d in psi do:
3       if d = 1
4            r = r ⊗ [0 1]
5       else if d = 0
6            r = r ⊗ [1 0]
7  return r
```

I(k)
1 $r = \begin{bmatrix}1\end{bmatrix}$
2 $id = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$
3 **while** $k \geq 1$ **do:**
4 $\quad r = r \otimes id$
5 $\quad k = k - 1$
6 **return** r

STEP(i, n)
1 $j = i$
2 $r = \begin{bmatrix}1\end{bmatrix}$
3 $id = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$
4 $p0 = \begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix}$
5 $p1 = \begin{bmatrix}0 & 0\\ 0 & 1\end{bmatrix}$
6 **while** $j \geq 2$ **do:**
7 $\quad r_j = \begin{bmatrix}1 & 0\\ 0 & \mathrm{e}^{\frac{2\pi \mathrm{i}}{2^j}}\end{bmatrix}$
8 $\quad r = r \cdot (\mathrm{I}(i) \otimes (id \otimes \mathrm{I}(j-2) \otimes p0 + r_j \otimes \mathrm{I}(j-2) \otimes p0) \otimes \mathrm{I}(n-i-j))$
9 $\quad j = j - 1$
10 **return** r

QTF(n)
1 $r = I(n)$
2 $had = \begin{bmatrix}\sqrt{0.5} & \sqrt{0.5}\\ \sqrt{0.5} & -\sqrt{0.5}\end{bmatrix}$
3 $i = n - 1$
4 **while** $i \geq 0$ **do:**
5 $\quad r = r \cdot \text{STEP}(n-i, n) \cdot (\mathrm{I}(i) \otimes had \otimes \mathrm{I}(n-i-1))$
6 $\quad i = i - 1$
7 **return** r

An example application could be $QFT(4) \cdot STATE(0010)$. Now we move on to show how we construct and apply the circuit using tensor networks.

3 Tensor Network Simulation

Tensors are mathematical objects with a lot in common with n-dimensional arrays. Some examples of tensors you are probably familliar with, a rank 0 tensor is just a scalar, a rank 1 tensor is a vector, a rank 2 tensor is a matrix, and so on. The rank of the tensor tells you how many indecies you need to describe a single element in the tensor, e.g. you need 1 index to tell which element in a vector you are talking about and you need 2, namely a column and row index to uniquely specify an element in a matrix. Each index has a dimension, for example a rank 2 tensor that is m by n would have one index i with dimmension n and another index j with dimension m, we can then label an element in the tensor a_{ij} where $i \in \{0, 1, .., n-1\}, j \in \{0, 1, .., m-1\}$. If we have more indecies they are sometimes written as a_{ij}^{kl}. We can use diagrams to draw these tensors, the tensors are repressented by red squares and the indecies are repressented by lines attaching to the square. The shape and colour of the tensors and the position of the lines typically do not matter, what is important is which of them are connected by shared indecies.

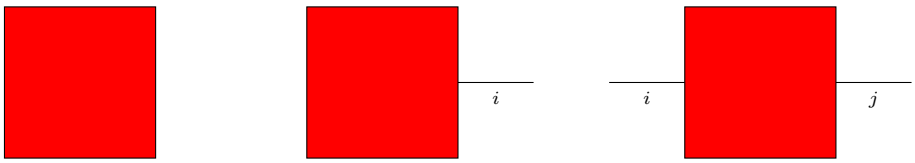

Figure 32: rank 0, 1 and 2 tensor. Each index is repressented by a line

Often the indecies are not labled on these drawings. If we have two tensors we can make them share an index if they each have an index with the same dimension. So if we have two rank 3 tensors, one where the dimensions of the indexes are $2, 3, 5$ and another where the dimensions are $1, 4, 3$, we can see that they both have an index of dimension 3 so if we wish we can make them share label for this index. Thus they could have the indecies i, j, k and l, m, j, notice the reuse of j.

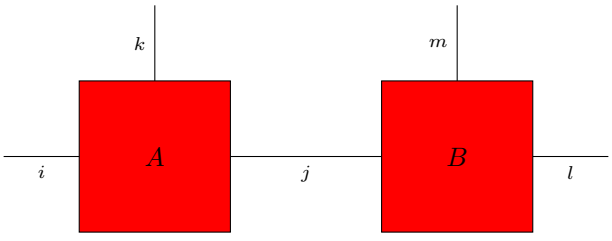

Figure 33: rank 3 tensors sharing index j.

3.1 Contracting tensors

But what is this good for? In essence, when we make tensors share indecies it is a way to symbolise that we are *contracting* them. The acctual computation of the contraction can be done at a later point we are just marking our intention. This is important as we might want to build an entire network of tensors and then be strategic about which shared indecies we choose to contract. Contracting a shared index is synonumus with computing the product of the two tensors.

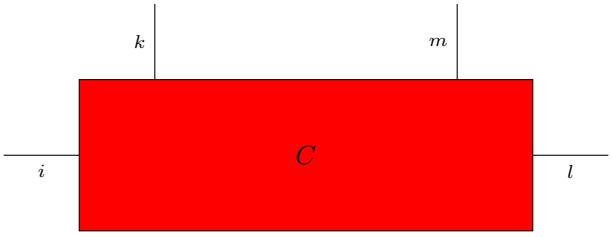

Figure 34: rank 4 tensor after contracting shared index j from Figure 33.

In this example we have two tensors A and B where each have elements of the form a_{ijk}, b_{lmj} resulting in a single tensor C with the elements

$$c_{iklm} = \sum_{x=0}^{dim(j)-1} a_{ixk} b_{lmx}$$

Here $dim(j) = 3$ and we subtract 1 since the elements are 0 indexed. In *tensor notation* the summation is implicit if there is a shared index in such an equation so we would just write $c_{iklm} = a_{ijk} b_{lmj}$

3.2 Reshaping tensors

$|\Psi\rangle$ is a vector, in other words a rank 1 tensor, but later we are using them as rank 2 and 3 tensors, how can we do so? We can simply add some ranks like so, $|\Psi\rangle \rightarrow \big[|\Psi\rangle\big] \rightarrow \big[\big[|\Psi\rangle\big]\big]$, this can be done for the gates as well, we just add indecies with dimension 1. The more general rule goes that you can change the rank of the tensor as much as you want as long as the product of the dimensions is the same. By this logic we can always add indecies with dimension 1, but we can also split indecies in two as long as the product of the dimensions of the new indecies equals the old indexs dimensions. We can also combine indecies. This operation is often called reshaping.

3.3 Splitting two qubit gates into two tensors with a shared index

The SVD[3] is a mathematical operation that disassebles a matrix A into a matrix U, a vector of singular values λ and a matrix V, which upholds the

property $A = U\Lambda Vh$ where Λ is a matrix where the diagonal corresponds to the values of λ and all the other elements are 0, and Vh is the conjugate transpose of V.

To take advatage of this we have our 4x4 gates, they have two indecies that each have a dimension 4 but we reshape to 4 indecies with dimensions 2 - an input for each qubit and an output for each. So if we describe our matrix elements with regards to these indecies we can say an element $a_{i_1 i_2 o_1 o_2}$ is in either the top half or the bottom half depending on i_1, and in the left half or right half depending on o_1 this gives us a sub matrix that is 2x2, which we divide the same way using i_2 and o_2. So our matrix elements are divided like this

$$\begin{bmatrix} a_{0000} & a_{0001} & a_{0010} & a_{0011} \\ a_{0100} & a_{0101} & a_{0110} & a_{0111} \\ a_{1000} & a_{1001} & a_{1010} & a_{1011} \\ a_{1100} & a_{1101} & a_{1110} & a_{1111} \end{bmatrix}$$

Figure 35: Matrix element indecies

Unfortunately the SVD splits vertically, i.e in this case we would be splitting the gate between the inputs and outputs, but we are interested in splitting between the two qubits, so i_1 and o_1 are seperate from i_2 and o_2. Luckily we can just swap around the indecies and thus elements beforehand, we just have to remember to swap them back when all is set and done.

$$\begin{bmatrix} a_{0000} & a_{0001} & a_{0100} & a_{0101} \\ a_{0010} & a_{0011} & a_{0110} & a_{0111} \\ a_{1000} & a_{1001} & a_{1100} & a_{1101} \\ a_{1010} & a_{1011} & a_{1110} & a_{1111} \end{bmatrix}$$

Figure 36: Indecies swapped

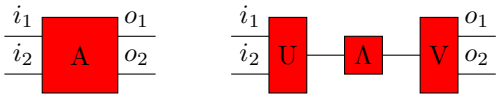

Figure 37: SVD whithout swapping

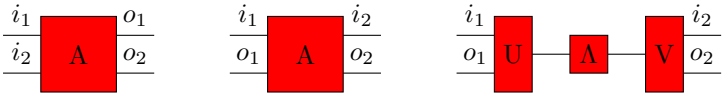

Figure 38: SVD whith swapping

We do though want to express this using two tensors only, but this is fortunately easy as we can just split Λ by squaring every value in it. Now we can contract two indecies and get the result we want.

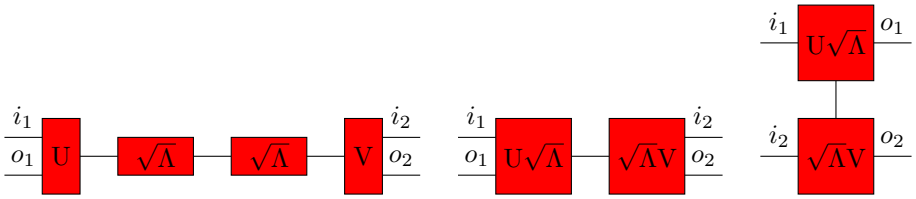

Figure 39: Combine the network from figure 38 into two tensors.

This whole process is sometimes called the operator Schmidt decomposition[4].

3.4 Repressenting a circuit and state using tensors

We can use a tensor network to represent our QFT circuit and our state. We will use a matrix product state (MPS) to represent our state, and a matrix product operator (MPO) to repressent our circuit, these are just sub categories of tensor networks.

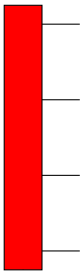

Figure 40: 4 qubit state repressented as a tensor. (MPS)

Here we have the whole state in one large tensor. This is highly undesirable as this uses a lot of memory, we would rather construct a network of tensors. Let us take the state $|0010\rangle$ as an example.

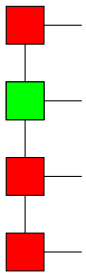

Figure 41: 4 qubit state $|0010\rangle$ represented as a tensor network (MPS). Red represents $|0\rangle$ and green $|1\rangle$.

Here we reshape the tensors repressenting each qubit to fit with our network structure. They simply have shared indecies of size 1 connecting them. Now let us look at repressenting the QFT as a MPO:

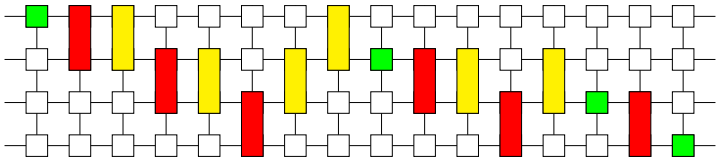

Figure 42: QFT_4 represented as a tensor network (MPO). White represents an identity gate, green a Hadamard gate, red a controlled $R_x, x \in \{2, 3, 4\}$ gate, and yellow a swap gate.

Now you might ask why we suddenly are using a ton of swap gates, and what are they even. Swap gates simply swap the values of two qubits, so if $|10\rangle$ goes in $|01\rangle$ commes out. This is simulated with this matrix, when swapping adjacent qubits as we are.

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

Figure 43: SWAP gate matrix.

We do this to avoid having non-local gates in the circuit, such as the controlled R_x gates we use in the QFT. This was not a concern when doing the dense simulation, but here we have to split everything up into tensors acting on individual qubits and therefor would like smaller gates to start with. The reason we have to split everything up is mostly a limitation of the liberary we are using for the simulation, it is a perfectly vallid tensor network even without splitting every

gate. It does add a cost of slightly less than two swap gates per non-local gate. This low overhead is not something that can be achieved in general, the most general way to make every gate local requres swapping the two qubits with the ones between them until they are adjacent, applying the gate, and then swapping them back to where they came from, this takes swapgates equal to the number of qubits between the two target qubits times two. We are fortunately in a situation where we can do this in a smarter way.

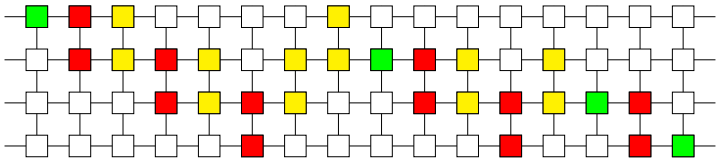

Figure 44: QFT_4 from figure 42 with all two qubit gates split using the operator Schmidt decomposition.

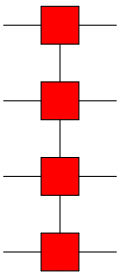

Figure 45: QFT_4 from figure 44 with 'horisontal' shared indecies contracted.

We have used the operator Schmidt decomposition to split the two qubit gates such that we can apply them to each of our qubits seperately. Now that we have our state MPS and our QFT MPO we can apply the MPO to the MPS by connecting the outputs from the MPS to the inputs of the MPO and contracting those shared indexes. This results in an updated MPS. If we wish to combine two MPOs into one we do the same, just connecting the output of one to the input of another and contracting.

(a) Before contraction. (b) After contraction.

Figure 46: MPO applied to MPS.

After splitting a gate which acts on more than one qubit the index they share will have a dimension of 1 or larger, and when we contract a MPO like so, we can see that the dimension scales quite quickly, with the layers if we do not do something:

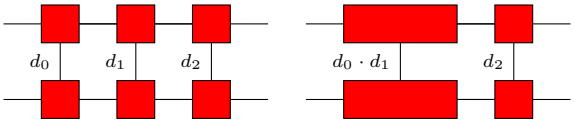

Figure 47: MPO with vertical bond dimensions d_i.

So what can we do? We can compress the bonds in our MPO every now and then, to get them back to a managable size, this is possible since we know our QFT has a fixed max bond dimension. The way we do it is to go though our bonds, e.g from top to bottom and contract then re-split them using the SVD. This will help, but we can do better if we are willing to pay a little accuracy, we can truncate the singular values to only the ones that contribute the most, and cut out those that do not contribute a lot to the result. In the QFT circuit it has been shown that the singular values decrese exponentially so we don't lose much precission.

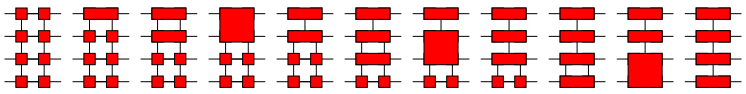

Figure 48: From left to right: Steps in compressing an MPO.

Compressing a MPO takes time, but frees up memory, it is therefor important to weigh these two against each other when implementing such a simulation, as we can get dramatic speedups by sacrificing a little memory, but due to the scaling of the bond dimensions we quickly end up having to compress to not run out of memory.

So far we have described how to take the vectors and matrix repressentations of gates and add dimmensions to them to make them fit our MPS and MPO repressentations. We show how to take a matrix for a two qubit gate and split it into two seperate matricies. We show how to contract a tensor network and how to compress an MPO after some amount of applications of other MPOs. We use this to repressent the QFT as a tensor network using only local gates, using swap gates in the construction.

4 Results & conclusion

During this project I implementet both the dense and tensor network simulations and here are some results, unless otherwise mentioned all tensor network simulations are run with max bond at 32.

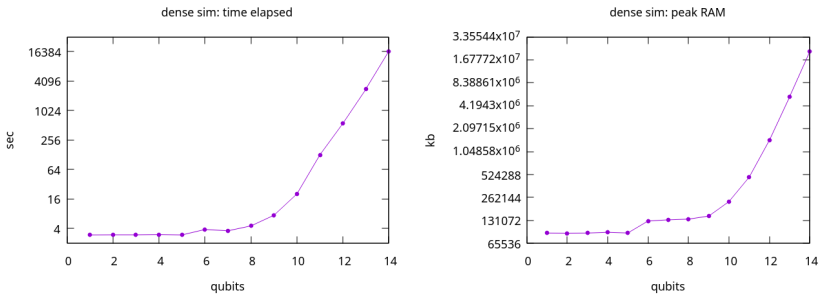

Figure 49: Time and peak RAM usage from constructing the dense matricies, using the approach discussed in section 2.

We see here that both time and memory usage scales faster that quadratic in the number of qubits simulated in the dense simulation. If we look at the tensor network simulation we see a different picture; time scales quadratically and memory has these spikes but seems to not be affected by the number of qubits after a while. We also notice that before we had to stop at 14 qubits now we arbitrarily stop at 32 qubits, but if we want we can run it with 64 qubits in around 30 minutes or larger if we have the time.

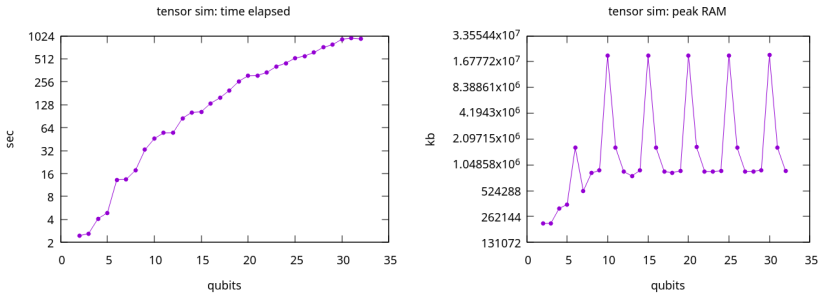

Figure 50: Time and peak RAM usage from constructing the MPOs, using the approach discussed in section 3, compressing after every fifth gate applied.

The spikes stem from only compressing the MPO every five gates, to confirm this we do the same benchmark where we compress every time.

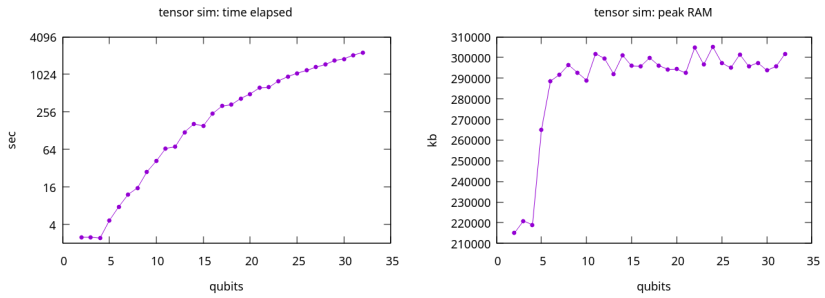

Figure 51: Time and peak RAM usage from constructing the MPOs, using the approach discussed in section 3, compressing after every gate applied.

Here we can see the spikes disappear, but we trade it for more time elapsed about 2.5x for 32 qubits. For 64 qubits with a max bond at 65 the difference is 30 minutes against 1 hour and 40 minutes, i.e. a 3.3x speedup gained by skipping some compressions.

Now for the precission, the whole project is built upon the resently proven claim that the singular values are exponentially decreasing in the QFT circuit and therefor we can get most of the precission with a low max bond on the network, here we see a plot of the precission for both the compress every time and every fifth time

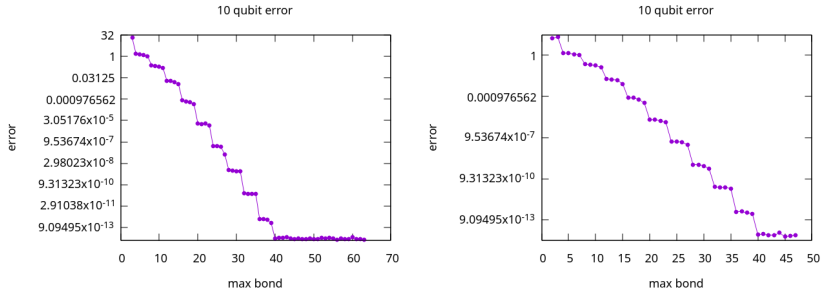

Figure 52: Error obtained by converting the MPO to its dense repressentation and taking the operator norm of the difference with the exact matrix produced by the dense simulation. Left: compress every time, Right: compress every fifth time.

We see here that the error is approximatly the same whether we compress every time or every fifth time. We also notice that it decreasses less and less and at some point completely stops improving as we increase the max bond. This is exactly as we expected to see. This is because at some point around a max bond of 40 we keep any relevant data for this circuit even after the compression.

If we look at how the memory and time usage is affected by adjusting the max bond we see that it is a powerfull lever for trading precission for performance.

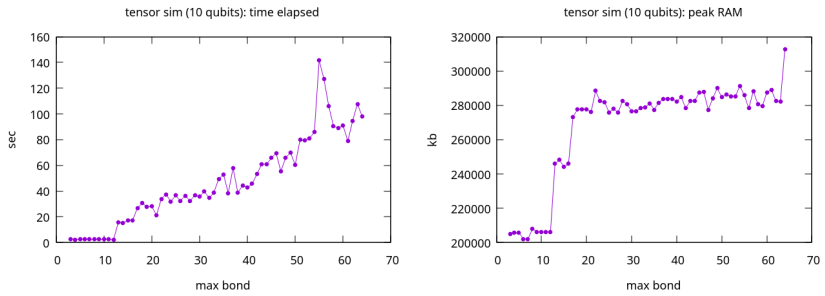

Figure 53: Time and peak RAM usage from constructing the MPOs, using the approach discussed in section 3, compressing after every gate applied.

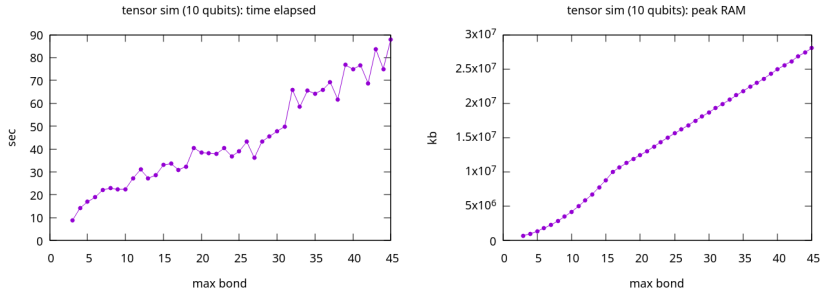

Figure 54: Time and peak RAM usage from constructing the MPOs, using the approach discussed in section 3, compressing after every fifth gate applied.

The reason we do not extend the graph for the benchmark where we compress every fifth time beyond 47 as max bond is because the spikes in memory usage scale with the max bond and the gap between compressions as seen in the peak memory graph in figure 54.

In conclusion I have implemented enough of the ideas from the paper to see the effects play out and it is indeed far more efficient to simulate the QFT using tensor networs, with close to no error.

5 Further work & acknowledgements

I unfortunatly did not have time to look into nor implement the zip-up algorithm discussed in the paper which could be someting intersting to work on. I would like to acknowledge Michael Kastoryano and Nutan Limaye for their supervision, I would not have been even a third of the way by now without them. I would also like to acknowledge Sebastian Loeschcke for giving a few tips and tricks for working with the python liberary quimb, which saved me many hours.

References

[1] Jielun Chen, E.M. Stoudenmire, and Steven R. White. The quantum fourier transform has small entanglement (arxiv:2210.08468v2 [quant-ph]). 2022.

[2] Simon Dirlik. A comparison of fft processor design. 2013.

[3] J. C. Nash and S. Shlien. Simple Algorithms for the Partial Singular Value Decomposition. *The Computer Journal*, 30(3):268–275, 01 1987.

[4] Jon E. Tyson. Operator-schmidt decompositions and the fourier transform, with applications to the operator-schmidt numbers of unitaries (arxiv:quant-ph/0306144v2). 2003.

6 Appendix

6.1 calculations for section 2.3

$$\mathbf{I}\,|\mathbf{0}\rangle = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}1\\ 0\end{bmatrix} = \begin{bmatrix}1\cdot 1 + 0\cdot 0\\ 0\cdot 1 + 1\cdot 0\end{bmatrix} = \begin{bmatrix}1\\ 0\end{bmatrix} = |\mathbf{0}\rangle$$
$$\mathbf{I}\,|\mathbf{1}\rangle = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}0\\ 1\end{bmatrix} = \begin{bmatrix}1\cdot 0 + 0\cdot 1\\ 0\cdot 0 + 1\cdot 1\end{bmatrix} = \begin{bmatrix}0\\ 1\end{bmatrix} = |\mathbf{1}\rangle$$

Figure 55: Applying $\mathbf{I}$ gate to the states $|\mathbf{0}\rangle$ and $|\mathbf{1}\rangle$

$$\mathbf{X}\,|\mathbf{0}\rangle = \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}\begin{bmatrix}1\\ 0\end{bmatrix} = \begin{bmatrix}0\cdot 1 + 1\cdot 0\\ 1\cdot 1 + 0\cdot 0\end{bmatrix} = \begin{bmatrix}0\\ 1\end{bmatrix} = |\mathbf{1}\rangle$$
$$\mathbf{X}\,|\mathbf{1}\rangle = \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}\begin{bmatrix}0\\ 1\end{bmatrix} = \begin{bmatrix}0\cdot 0 + 1\cdot 1\\ 1\cdot 0 + 0\cdot 1\end{bmatrix} = \begin{bmatrix}1\\ 0\end{bmatrix} = |\mathbf{0}\rangle$$

Figure 56: Applying $\mathbf{X}$ gate to the states $|\mathbf{0}\rangle$ and $|\mathbf{1}\rangle$

$$\mathbf{H}\,|\mathbf{0}\rangle = \begin{bmatrix}\sqrt{0.5} & \sqrt{0.5}\\ \sqrt{0.5} & -\sqrt{0.5}\end{bmatrix}\begin{bmatrix}1\\ 0\end{bmatrix} = \begin{bmatrix}\sqrt{0.5}\cdot 1 + \sqrt{0.5}\cdot 0\\ \sqrt{0.5}\cdot 1 - \sqrt{0.5}\cdot 0\end{bmatrix} = \begin{bmatrix}\sqrt{0.5}\\ \sqrt{0.5}\end{bmatrix} = |+\rangle$$
$$\mathbf{H}\,|\mathbf{1}\rangle = \begin{bmatrix}\sqrt{0.5} & \sqrt{0.5}\\ \sqrt{0.5} & -\sqrt{0.5}\end{bmatrix}\begin{bmatrix}0\\ 1\end{bmatrix} = \begin{bmatrix}\sqrt{0.5}\cdot 0 + \sqrt{0.5}\cdot 1\\ \sqrt{0.5}\cdot 0 - \sqrt{0.5}\cdot 1\end{bmatrix} = \begin{bmatrix}\sqrt{0.5}\\ -\sqrt{0.5}\end{bmatrix} = |-\rangle$$

Figure 57: Applying $\mathbf{H}$ gate to the states $|\mathbf{0}\rangle$ and $|\mathbf{1}\rangle$

$$\mathbf{R_2}\,|\mathbf{0}\rangle = \begin{bmatrix}1 & 0\\ 0 & \mathrm{e}^{\frac{2\pi\mathrm{i}}{2^2}}\end{bmatrix}\begin{bmatrix}1\\ 0\end{bmatrix} = \begin{bmatrix}1\cdot 1 + 0\cdot 0\\ 0\cdot 1 + \mathrm{e}^{\frac{2\pi\mathrm{i}}{2^2}}\cdot 0\end{bmatrix} = \begin{bmatrix}1\\ 0\end{bmatrix} = |\mathbf{0}\rangle$$
$$\mathbf{R_2}\,|\mathbf{1}\rangle = \begin{bmatrix}1 & 0\\ 0 & \mathrm{e}^{\frac{2\pi\mathrm{i}}{2^2}}\end{bmatrix}\begin{bmatrix}0\\ 1\end{bmatrix} = \begin{bmatrix}1\cdot 0 + 0\cdot 1\\ 0\cdot 0 + \mathrm{e}^{\frac{2\pi\mathrm{i}}{2^2}}\cdot 1\end{bmatrix} = \begin{bmatrix}0\\ \mathrm{e}^{\frac{2\pi\mathrm{i}}{2^2}}\end{bmatrix} = \begin{bmatrix}0\\ \mathrm{i}\end{bmatrix}$$

Figure 58: Applying $\mathbf{R_2}$ gate to the states $|\mathbf{0}\rangle$ and $|\mathbf{1}\rangle$

$$\mathbf{CX}\,|\mathbf{00}\rangle = \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}\begin{bmatrix}1\\0\\0\\0\end{bmatrix} = \begin{bmatrix}1\cdot 1+0\cdot 0+0\cdot 0+0\cdot 0\\0\cdot 1+1\cdot 0+0\cdot 0+0\cdot 0\\0\cdot 1+0\cdot 0+0\cdot 0+1\cdot 0\\0\cdot 1+0\cdot 0+1\cdot 0+0\cdot 0\end{bmatrix} = \begin{bmatrix}1\\0\\0\\0\end{bmatrix} = |\mathbf{00}\rangle$$

$$\mathbf{CX}\,|\mathbf{01}\rangle = \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}\begin{bmatrix}0\\1\\0\\0\end{bmatrix} = \begin{bmatrix}1\cdot 0+0\cdot 1+0\cdot 0+0\cdot 0\\0\cdot 0+1\cdot 1+0\cdot 0+0\cdot 0\\0\cdot 0+0\cdot 1+0\cdot 0+1\cdot 0\\0\cdot 0+0\cdot 1+1\cdot 0+0\cdot 0\end{bmatrix} = \begin{bmatrix}0\\1\\0\\0\end{bmatrix} = |\mathbf{01}\rangle$$

$$\mathbf{CX}\,|\mathbf{10}\rangle = \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}\begin{bmatrix}0\\0\\1\\0\end{bmatrix} = \begin{bmatrix}1\cdot 0+0\cdot 0+0\cdot 1+0\cdot 0\\0\cdot 0+1\cdot 0+0\cdot 1+0\cdot 0\\0\cdot 0+0\cdot 0+0\cdot 1+1\cdot 0\\0\cdot 0+0\cdot 0+1\cdot 1+0\cdot 0\end{bmatrix} = \begin{bmatrix}0\\0\\0\\1\end{bmatrix} = |\mathbf{11}\rangle$$

$$\mathbf{CX}\,|\mathbf{11}\rangle = \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}\begin{bmatrix}0\\0\\0\\1\end{bmatrix} = \begin{bmatrix}1\cdot 0+0\cdot 0+0\cdot 0+0\cdot 1\\0\cdot 0+1\cdot 0+0\cdot 0+0\cdot 1\\0\cdot 0+0\cdot 0+0\cdot 0+1\cdot 1\\0\cdot 0+0\cdot 0+1\cdot 0+0\cdot 1\end{bmatrix} = \begin{bmatrix}0\\0\\1\\0\end{bmatrix} = |\mathbf{10}\rangle$$

Figure 59: Applying **CX** gate to the states $|\mathbf{00}\rangle$, $|\mathbf{01}\rangle$, $|\mathbf{10}\rangle$ and $|\mathbf{11}\rangle$

6.2 calculations for section 2.4

$$(\mathbf{I}\otimes\mathbf{I}\otimes\mathbf{X})\,|\mathbf{000}\rangle = \left(\begin{bmatrix}1&0\\0&1\end{bmatrix}\otimes\begin{bmatrix}1&0\\0&1\end{bmatrix}\otimes\begin{bmatrix}0&1\\1&0\end{bmatrix}\right)\left(\begin{bmatrix}1\\0\end{bmatrix}\otimes\begin{bmatrix}1\\0\end{bmatrix}\otimes\begin{bmatrix}1\\0\end{bmatrix}\right) =$$

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

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

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

$$\begin{bmatrix}0\cdot1+1\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0\\1\cdot1+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0\\0\cdot1+0\cdot0+0\cdot0+1\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0\\0\cdot1+0\cdot0+1\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0\\0\cdot1+0\cdot0+0\cdot0+0\cdot0+0\cdot0+1\cdot0+0\cdot0+0\cdot0\\0\cdot1+0\cdot0+0\cdot0+0\cdot0+1\cdot0+0\cdot0+0\cdot0+0\cdot0\\0\cdot1+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+1\cdot0+0\cdot0\\0\cdot1+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+1\cdot0\end{bmatrix} = \begin{bmatrix}0\\1\\0\\0\\0\\0\\0\\0\end{bmatrix} = |\mathbf{001}\rangle$$

Figure 60: Applying $\mathbf{X}$ to the first qubit in a 3 qubit system

$$(\mathbf{I}\otimes\mathbf{X}\otimes\mathbf{I})\,|\mathbf{000}\rangle = \left(\begin{bmatrix}1&0\\0&1\end{bmatrix}\otimes\begin{bmatrix}0&1\\1&0\end{bmatrix}\otimes\begin{bmatrix}1&0\\0&1\end{bmatrix}\right)\left(\begin{bmatrix}1\\0\end{bmatrix}\otimes\begin{bmatrix}1\\0\end{bmatrix}\otimes\begin{bmatrix}1\\0\end{bmatrix}\right) =$$

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

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

$$\begin{bmatrix}0\cdot1+0\cdot0+1\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0\\0\cdot1+0\cdot0+0\cdot0+1\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0\\1\cdot1+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0\\0\cdot1+1\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0\\0\cdot1+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+1\cdot0+0\cdot0\\0\cdot1+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+1\cdot0\\0\cdot1+0\cdot0+0\cdot0+0\cdot0+1\cdot0+0\cdot0+0\cdot0+0\cdot0\\0\cdot1+0\cdot0+0\cdot0+0\cdot0+0\cdot0+1\cdot0+0\cdot0+0\cdot0\end{bmatrix} = \begin{bmatrix}0\\0\\1\\0\\0\\0\\0\\0\end{bmatrix} = |\mathbf{010}\rangle$$

Figure 61: Applying $\mathbf{X}$ to the second qubit in a 3 qubit system

$$(\mathbf{X}\otimes\mathbf{I}\otimes\mathbf{I})\,|\mathbf{000}\rangle = \left(\begin{bmatrix}0&1\\1&0\end{bmatrix}\otimes\begin{bmatrix}1&0\\0&1\end{bmatrix}\otimes\begin{bmatrix}1&0\\0&1\end{bmatrix}\right)\left(\begin{bmatrix}1\\0\end{bmatrix}\otimes\begin{bmatrix}1\\0\end{bmatrix}\otimes\begin{bmatrix}1\\0\end{bmatrix}\right) =$$

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

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

$$\begin{bmatrix}0\cdot1+0\cdot0+0\cdot0+0\cdot0+1\cdot0+0\cdot0+0\cdot0+0\cdot0\\0\cdot1+0\cdot0+0\cdot0+0\cdot0+0\cdot0+1\cdot0+0\cdot0+0\cdot0\\0\cdot1+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+1\cdot0+0\cdot0\\0\cdot1+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+1\cdot0\\1\cdot1+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0\\0\cdot1+1\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0\\0\cdot1+0\cdot0+1\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0\\0\cdot1+0\cdot0+0\cdot0+1\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0\end{bmatrix} = \begin{bmatrix}0\\0\\0\\0\\1\\0\\0\\0\end{bmatrix} = |\mathbf{100}\rangle$$

Figure 62: Applying $\mathbf{X}$ to the thrid qubit in a 3 qubit system

If we want to apply multiple gates at a time we just replace the corresponding **I** gates.

$$(\mathbf{X}\otimes\mathbf{I}\otimes\mathbf{H})\,|\mathbf{000}\rangle = \left(\begin{bmatrix}0&1\\1&0\end{bmatrix}\otimes\begin{bmatrix}1&0\\0&1\end{bmatrix}\otimes\begin{bmatrix}\sqrt{0.5}&\sqrt{0.5}\\\sqrt{0.5}&-\sqrt{0.5}\end{bmatrix}\right)\left(\begin{bmatrix}1\\0\end{bmatrix}\otimes\begin{bmatrix}1\\0\end{bmatrix}\otimes\begin{bmatrix}1\\0\end{bmatrix}\right) =$$

$$\begin{bmatrix}
0\begin{bmatrix}1\begin{bmatrix}\sqrt{0.5}&\sqrt{0.5}\\\sqrt{0.5}&-\sqrt{0.5}\end{bmatrix}&0\begin{bmatrix}\sqrt{0.5}&\sqrt{0.5}\\\sqrt{0.5}&-\sqrt{0.5}\end{bmatrix}\\0\begin{bmatrix}\sqrt{0.5}&\sqrt{0.5}\\\sqrt{0.5}&-\sqrt{0.5}\end{bmatrix}&1\begin{bmatrix}\sqrt{0.5}&\sqrt{0.5}\\\sqrt{0.5}&-\sqrt{0.5}\end{bmatrix}\end{bmatrix} & 1\begin{bmatrix}1\begin{bmatrix}\sqrt{0.5}&\sqrt{0.5}\\\sqrt{0.5}&-\sqrt{0.5}\end{bmatrix}&0\begin{bmatrix}\sqrt{0.5}&\sqrt{0.5}\\\sqrt{0.5}&-\sqrt{0.5}\end{bmatrix}\\0\begin{bmatrix}\sqrt{0.5}&\sqrt{0.5}\\\sqrt{0.5}&-\sqrt{0.5}\end{bmatrix}&1\begin{bmatrix}\sqrt{0.5}&\sqrt{0.5}\\\sqrt{0.5}&-\sqrt{0.5}\end{bmatrix}\end{bmatrix}\\
1\begin{bmatrix}1\begin{bmatrix}\sqrt{0.5}&\sqrt{0.5}\\\sqrt{0.5}&-\sqrt{0.5}\end{bmatrix}&0\begin{bmatrix}\sqrt{0.5}&\sqrt{0.5}\\\sqrt{0.5}&-\sqrt{0.5}\end{bmatrix}\\0\begin{bmatrix}\sqrt{0.5}&\sqrt{0.5}\\\sqrt{0.5}&-\sqrt{0.5}\end{bmatrix}&1\begin{bmatrix}\sqrt{0.5}&\sqrt{0.5}\\\sqrt{0.5}&-\sqrt{0.5}\end{bmatrix}\end{bmatrix} & 0\begin{bmatrix}1\begin{bmatrix}\sqrt{0.5}&\sqrt{0.5}\\\sqrt{0.5}&-\sqrt{0.5}\end{bmatrix}&0\begin{bmatrix}\sqrt{0.5}&\sqrt{0.5}\\\sqrt{0.5}&-\sqrt{0.5}\end{bmatrix}\\0\begin{bmatrix}\sqrt{0.5}&\sqrt{0.5}\\\sqrt{0.5}&-\sqrt{0.5}\end{bmatrix}&1\begin{bmatrix}\sqrt{0.5}&\sqrt{0.5}\\\sqrt{0.5}&-\sqrt{0.5}\end{bmatrix}\end{bmatrix}
\end{bmatrix}\begin{bmatrix}1\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} =$$

$$\begin{bmatrix}
0&0&0&0&\sqrt{0.5}&\sqrt{0.5}&0&0\\
0&0&0&0&\sqrt{0.5}&-\sqrt{0.5}&0&0\\
0&0&0&0&0&0&\sqrt{0.5}&\sqrt{0.5}\\
0&0&0&0&0&0&\sqrt{0.5}&-\sqrt{0.5}\\
\sqrt{0.5}&\sqrt{0.5}&0&0&0&0&0&0\\
\sqrt{0.5}&-\sqrt{0.5}&0&0&0&0&0&0\\
0&0&\sqrt{0.5}&\sqrt{0.5}&0&0&0&0\\
0&0&\sqrt{0.5}&-\sqrt{0.5}&0&0&0&0
\end{bmatrix}\begin{bmatrix}1\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} =$$

$$\begin{bmatrix}
0\cdot1+0\cdot0+0\cdot0+0\cdot0+\sqrt{0.5}\cdot0+\sqrt{0.5}\cdot0+0\cdot0+0\cdot0\\
0\cdot1+0\cdot0+0\cdot0+0\cdot0+\sqrt{0.5}\cdot0-\sqrt{0.5}\cdot0+0\cdot0+0\cdot0\\
0\cdot1+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+\sqrt{0.5}\cdot0+\sqrt{0.5}\cdot0\\
0\cdot1+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+\sqrt{0.5}\cdot0-\sqrt{0.5}\cdot0\\
\sqrt{0.5}\cdot1+\sqrt{0.5}\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0\\
\sqrt{0.5}\cdot1-\sqrt{0.5}\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0\\
0\cdot1+0\cdot0+\sqrt{0.5}\cdot0+\sqrt{0.5}\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0\\
0\cdot1+0\cdot0+\sqrt{0.5}\cdot0-\sqrt{0.5}\cdot0+0\cdot0+0\cdot0+0\cdot0+0\cdot0
\end{bmatrix} = \begin{bmatrix}0\\0\\0\\0\\\sqrt{0.5}\\\sqrt{0.5}\\0\\0\end{bmatrix} = |\mathbf{10+}\rangle$$

Figure 63: Applying **X** to the thrid qubit and **H** to the first qubit in a 3 qubit system

$$\mathbf{U} = \mathbf{CX}\,(\mathbf{I} \otimes \mathbf{H})\,(\mathbf{I} \otimes \mathbf{X}) =$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \begin{bmatrix} \sqrt{0.5} & \sqrt{0.5} \\ \sqrt{0.5} & -\sqrt{0.5} \end{bmatrix} & 0 \begin{bmatrix} \sqrt{0.5} & \sqrt{0.5} \\ \sqrt{0.5} & -\sqrt{0.5} \end{bmatrix} \\ 0 \begin{bmatrix} \sqrt{0.5} & \sqrt{0.5} \\ \sqrt{0.5} & -\sqrt{0.5} \end{bmatrix} & 1 \begin{bmatrix} \sqrt{0.5} & \sqrt{0.5} \\ \sqrt{0.5} & -\sqrt{0.5} \end{bmatrix} \end{bmatrix} \begin{bmatrix} 1 \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} & 0 \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \\ 0 \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} & 1 \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \end{bmatrix} =$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \sqrt{0.5} & \sqrt{0.5} & 0 & 0 \\ \sqrt{0.5} & -\sqrt{0.5} & 0 & 0 \\ 0 & 0 & \sqrt{0.5} & \sqrt{0.5} \\ 0 & 0 & \sqrt{0.5} & -\sqrt{0.5} \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} =$$

$$\begin{bmatrix} \sqrt{0.5} & \sqrt{0.5} & 0 & 0 \\ \sqrt{0.5} & -\sqrt{0.5} & 0 & 0 \\ 0 & 0 & \sqrt{0.5} & -\sqrt{0.5} \\ 0 & 0 & \sqrt{0.5} & \sqrt{0.5} \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} =$$

$$\begin{bmatrix} \sqrt{0.5} & \sqrt{0.5} & 0 & 0 \\ -\sqrt{0.5} & \sqrt{0.5} & 0 & 0 \\ 0 & 0 & -\sqrt{0.5} & \sqrt{0.5} \\ 0 & 0 & \sqrt{0.5} & \sqrt{0.5} \end{bmatrix}$$

Figure 64: Computing a matrix representation of a circuit,**U**, that first applies **X** then **H** to the first qubit then **CX** to the first and second qubit

$$\mathbf{U}\,|\mathbf{00}\rangle = \begin{bmatrix} \sqrt{0.5} & \sqrt{0.5} & 0 & 0 \\ -\sqrt{0.5} & \sqrt{0.5} & 0 & 0 \\ 0 & 0 & -\sqrt{0.5} & \sqrt{0.5} \\ 0 & 0 & \sqrt{0.5} & \sqrt{0.5} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \sqrt{0.5} \\ -\sqrt{0.5} \\ 0 \\ 0 \end{bmatrix} = |\mathbf{0}-\rangle$$

$$\mathbf{U}\,|\mathbf{01}\rangle = \begin{bmatrix} \sqrt{0.5} & \sqrt{0.5} & 0 & 0 \\ -\sqrt{0.5} & \sqrt{0.5} & 0 & 0 \\ 0 & 0 & -\sqrt{0.5} & \sqrt{0.5} \\ 0 & 0 & \sqrt{0.5} & \sqrt{0.5} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \sqrt{0.5} \\ \sqrt{0.5} \\ 0 \\ 0 \end{bmatrix} = |\mathbf{0}+\rangle$$

$$\mathbf{U}\,|\mathbf{10}\rangle = \begin{bmatrix} \sqrt{0.5} & \sqrt{0.5} & 0 & 0 \\ -\sqrt{0.5} & \sqrt{0.5} & 0 & 0 \\ 0 & 0 & -\sqrt{0.5} & \sqrt{0.5} \\ 0 & 0 & \sqrt{0.5} & \sqrt{0.5} \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -\sqrt{0.5} \\ \sqrt{0.5} \end{bmatrix} = -\,|\mathbf{1}-\rangle$$

$$\mathbf{U}\,|\mathbf{11}\rangle = \begin{bmatrix} \sqrt{0.5} & \sqrt{0.5} & 0 & 0 \\ -\sqrt{0.5} & \sqrt{0.5} & 0 & 0 \\ 0 & 0 & -\sqrt{0.5} & \sqrt{0.5} \\ 0 & 0 & \sqrt{0.5} & \sqrt{0.5} \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \sqrt{0.5} \\ \sqrt{0.5} \end{bmatrix} = |\mathbf{1}+\rangle$$

Figure 65: Applying the circuit represented by the matrix **U** from figure 64 to different states