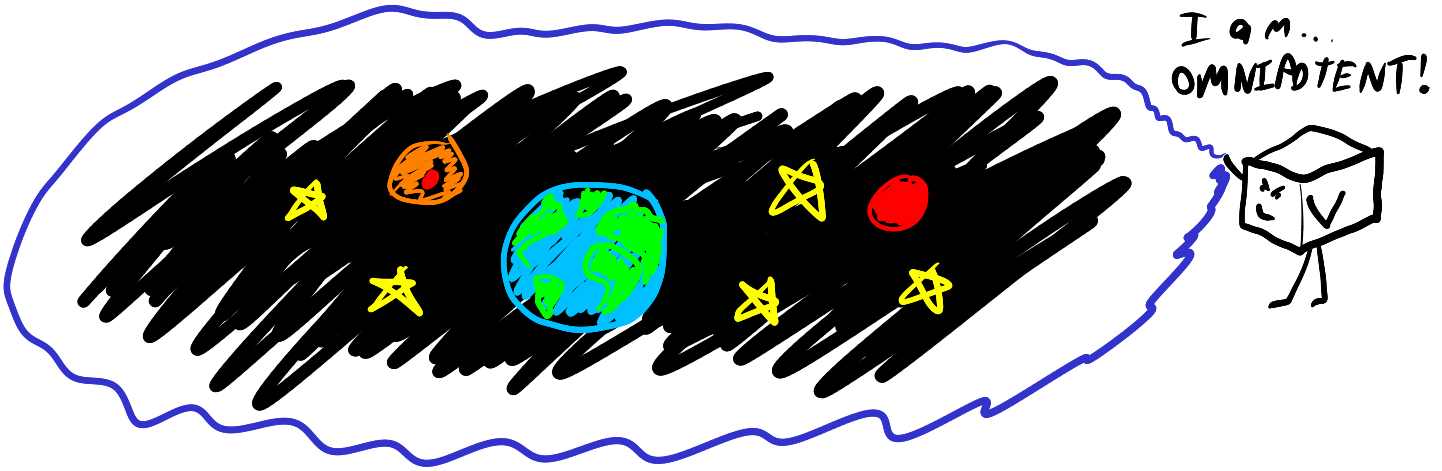
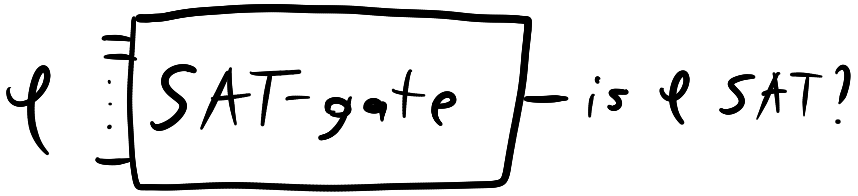
CMPT 476 Lecture 14

Gate sets & quantum universality

What does it mean to take e^T, where T is an operator?



In our previous discussion of the quantum circuit model, we made no mention of the gateset. However, in the classical circuit model, if our gateset consists of every classical function, then all complexity classes would collapse into constant time!



In the quantum circuit model, the situation is even more extreme — allowing any unitary matrix as a gate gives all "Turing degrees" in polynomial time!

To avoid problems like these, we typically restrict circuit models to a finite universal gateset. But is there such a gateset for quantum computing?

Read on to find out...

(A compendium of common gates)

Pauli gates:

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

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$ bit flip

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$ phase flip

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$$ bit & phase flip

Clifford gates (includes Pauli gates):

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$ Hadamard or branch gate

$$S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$$ phase gate

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$ controlled-NOT or quantum XOR

Cliffor gates are not universal.

There are a finite sets of unitaries generated by the Clifford gates.

Gottesman-Knill theorem:

Circuits consisting of only Clifford gates can be efficiently simulated classically

(And in particular are not universal)

Circuits over the clifford group are poly-time simulable

If this was universal for quantum computers, then

Non-Clifford gates:

$$T = \sqrt{S} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}$$

If you multiply e^(ipi/4) by itself, you will get i

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

Rotation gates:

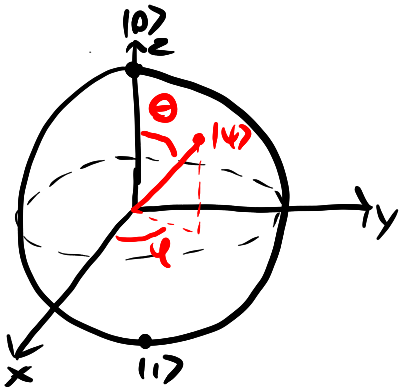
$$R_z(\theta) = \begin{bmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{bmatrix}$$ z rotation

$$R_x(\theta) = \begin{bmatrix} \cos\theta/2 & i\sin\theta/2 \\ -i\sin\theta/2 & \cos\theta/2 \end{bmatrix}$$ x rotation

$$R_y(\theta) = \begin{bmatrix} \cos\theta/2 & -\sin\theta/2 \\ \sin\theta/2 & \cos\theta/2 \end{bmatrix}$$ y rotation

(Single qubit gates & the Bloch sphere)

Recall that the state of a qubit is a vector on the Bloch sphere



Since a single-qubit unitary maps points on the Bloch sphere to points on the Bloch sphere in a reversible manner, every single-qubit unitary is a rotation of the Bloch sphere.

We can take every rotation around an axis and have an equivalent rotation around two non parallel axis.

Ex.

The X gate is a rotation around the x-axis of 180°.
It maps $|0\rangle \rightarrow |1\rangle$ and $|1\rangle \rightarrow |0\rangle$.

(Rotation gates)

The rotation gates $R_x(\theta)$, $R_z(\theta)$, $R_y(\theta)$ are rotations of angle θ around the x, z, and y axes respectively. They arise as matrix exponentials of Pauli gates:

$$R_x(\theta) = e^{-i\theta/2 X}$$

$$R_z(\theta) = e^{-i\theta/2 Z}$$

$$R_y(\theta) = e^{-i\theta/2 Y}$$

What does this mean?

(Important aside: operator functions)

Let $f: \mathbb{C} \to \mathbb{C}$ and $A: \mathcal{H} \to \mathcal{H}$ be defined as

$$A = \sum_i a_i |e_i\rangle\langle e_i|$$

for some orthonormal basis $\{|e_i\rangle\}$ of $\mathcal{H}$. We say A is diagonal in the basis $\{|e_i\rangle\}$, and

$$f(A) = \sum_i f(a_i)|e_i\rangle\langle e_i|$$

Ex.

$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$ is diagonal in the computational basis, and specifically

$$Z = |0\rangle\langle 0| - |1\rangle\langle 1|$$

Then

$$\begin{aligned} \sqrt{Z} &= \sqrt{1}|0\rangle\langle 0| + \sqrt{-1}|1\rangle\langle 1| \\ &= |0\rangle\langle 0| + i|1\rangle\langle 1| \\ &= \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \\ &= S \end{aligned}$$

Likewise,

Should the outer products not be in the exponent?
If not,

$$\begin{aligned} R_Z(\theta) = e^{-i\theta/2 Z} &= e^{-i\theta/2}|0\rangle\langle 0| + e^{i\theta/2}|1\rangle\langle 1| \\ &= \begin{bmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{bmatrix} \end{aligned}$$

Ask professor about this

What if A is not diagonal in the computational basis?

(The spectral theorem, for real this time)

An operator $A: \mathcal{H} \to \mathcal{H}$ is **normal** if & only if

$$AA^\dagger = A^\dagger A$$

All unitary or Hermitian operators are normal (why?)

The **Spectral theorem** states that every (finite dimensional) normal operator A can be written as

$$A = P \Lambda P^\dagger$$

Where:

1. Λ is diagonal as a matrix and encodes the **eigenvalues** of A
2. P is unitary and has as columns unit **eigenvectors** of A

we say that A **is diagonalizable** and

$$A = \sum_i \lambda_i P|i\rangle\langle i| P^\dagger$$

the **diagonalization** of A.

Ex.

Recall that X has eigenvectors $|+\rangle, |-\rangle$:

$$X|+\rangle = X\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right) = \frac{1}{\sqrt{2}}|1\rangle + \frac{1}{\sqrt{2}}|0\rangle = |+\rangle$$
$$X|-\rangle = X\left(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\right) = \frac{1}{\sqrt{2}}|1\rangle - \frac{1}{\sqrt{2}}|0\rangle = -|-\rangle$$

Since $H = [|+\rangle \; |-\rangle]$, H **diagonalizes** X

$$X = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} = HZH$$

(Back to exponentials)

Observe that if $A = P \Lambda P^\dagger$ is normal, then

$$f(A) = \sum_i f(\lambda_i) |i\rangle\langle i| P^\dagger$$
$$= P \left(\sum_i f(\lambda_i) |i\rangle\langle i| \right) P^\dagger$$
$$= P f(\Lambda) P^\dagger$$

Ex.

Since $X = HZH$, $R_X(\theta) = H e^{-i\theta/2 Z} H = H R_Z(\theta) H$.

A tedious calculation would show that

$$R_X(\theta) = \begin{bmatrix} \cos\theta/2 & -\sin\theta/2 \\ \sin\theta/2 & \cos\theta/2 \end{bmatrix}$$

Ask professor to go through the calculations

(A note on exponentials)

We won't make use of matrix exponentials often, but they are extremely important to Hamiltonian simulation, so it's useful to know and get comfortable with.

(Euler angle decomposition)

Let U be a single-qubit unitary. Then there exist angles $\alpha, \beta, \gamma, \delta \in \mathbb{R}$ such that

We want to reduce our gate to a finite gate set

$$U = e^{i\alpha} R_Z(\beta) R_X(\gamma) R_Z(\delta)$$

When we take the exponential of a x gate for a qudit, that is an analogue to rotating around the x-axis.

The above works with any other non-parallel axes as well, like Z & Y. This is a quantum re-statement of the very old fact that any rotation in 3-d space can be implemented as a sequence of 3 rotations in 2 planes.

SU(2) = SO(3) by an isomorphism
Unitary vec
We are working in C^4 since we have two complex numbers to work with

Representation theory is important

Single qubit is a 2x1 vector in c^2.

(Multi-qubit gates)

Since single-qubit gates can't entangle qubits, we know we'll need at least one multi-qubit gate for universal quantum computing.

One possibility is the SWAP gate, which we haven't seen before:

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

The SWAP gate swaps two qubits, and hence is not entangling.

Since we need an entangling gate, the swap gate will not suffice

$$\text{SWAP}\left(\begin{bmatrix} \alpha \\ \beta \end{bmatrix} \otimes \begin{bmatrix} \gamma \\ \delta \end{bmatrix}\right) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha\gamma \\ \alpha\delta \\ \beta\gamma \\ \beta\delta \end{bmatrix}$$

$$= \begin{bmatrix} \alpha\gamma \\ \beta\gamma \\ \alpha\delta \\ \beta\delta \end{bmatrix}$$

$$= \begin{bmatrix} \gamma \\ \delta \end{bmatrix} \otimes \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

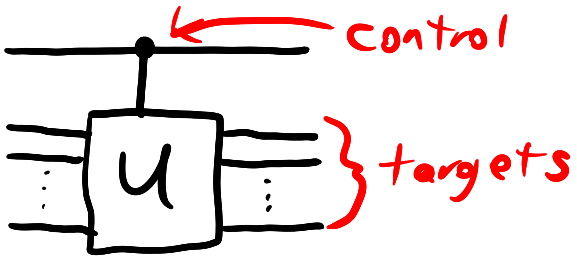
On the other hand, controlled gates usually are entangling. We've seen one such gate already: the CNOT gate.

(Controlled gates)

Let U be an n-qubit unitary. The controlled-U gate $c\text{-}U$ is an $n+1$-qubit unitary such that

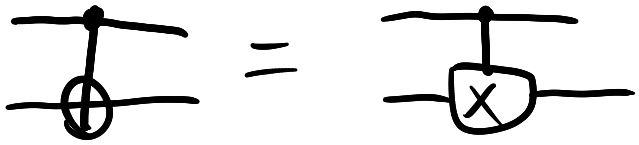
$$c\text{-}U|0\rangle|\psi\rangle = |0\rangle|\psi\rangle$$
$$c\text{-}U|1\rangle|\psi\rangle = |1\rangle(U|\psi\rangle)$$

We draw a controlled-U gate as



Ex.

The CNOT gate is the controlled-NOT, i.e.



It sends $|0\rangle|x\rangle \longmapsto |0\rangle|x\rangle$ and $|1\rangle|x\rangle \longmapsto |1\rangle|1 \oplus x\rangle$, or more concisely

$$\text{CNOT}: |x\rangle|y\rangle \longmapsto |x\rangle|x \oplus y\rangle$$

Ex.

The Toffoli gate is also a controlled gate,

$$\text{Toffoli} = c\text{-CNOT}$$

In particular, noting that

$$\text{Toffoli}\,|x\rangle|y\rangle|z\rangle = |x\rangle|y\rangle|z \oplus xy\rangle$$

we have

$$\begin{aligned}\text{Toffoli}\,|0\rangle|y\rangle|z\rangle &= |0\rangle|y\rangle|z\rangle \\ \text{Toffoli}\,|1\rangle|y\rangle|z\rangle &= |1\rangle|y\rangle|z \oplus y\rangle \\ &= |1\rangle(\text{CNOT}|y\rangle|z\rangle)\end{aligned}$$

When we define the control bit, where it depends on the topmost bit, then the control bit is always on the top.

Ex.

The controlled-Z or CZ gate is sometimes drawn

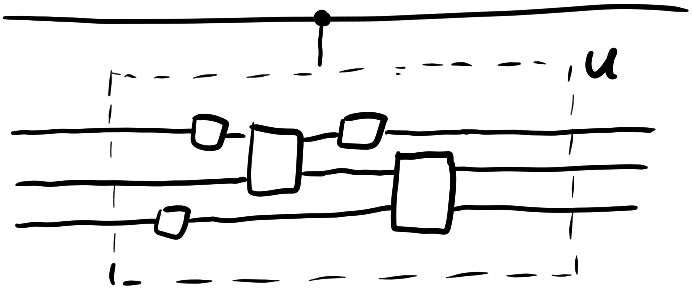
and has matrix

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

(Controlled gate implementations)

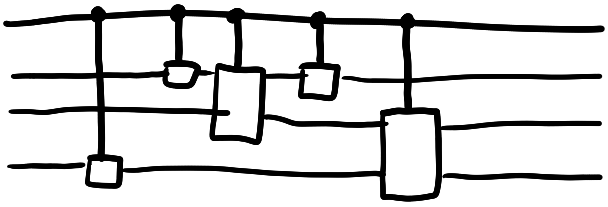
A common pattern in quantum computation is to take some unitary implemented as a circuit and control it.

When we want to compile quantum algorithms, we oftern want to control a specific gate set.



We can do this by controlling each gate in U individually

If the value of the first bit is 0, do none of the operations
If the value of the first bit is 1, then do all of the operations.

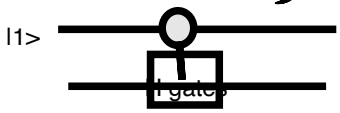


What if we can't decompose U further?

ie, what if we have a rotation matrix along the x, y, and z axis?

(Theorem, Barenco et. al.)

Every controlled unitary can be implemented as a circuit consisting of CNOT and single-qubit unitaries



If you have a controlled unitary, then you can also do the unitary by setting the ancilla to one

(Corollary)

{CNOT} + Single-qubit unitaries is universal for quantum computing.

This gate set is not physically implementable.
We apply a magnetic field for a specific time interval, and that will determine the angle you rotate by.
If we reset ourselves to a discrete set of angles and a finite set of gates, then we know we can only implement a countable number of unitaries.
We have an infinite number of unitaries over the complex field.

The proof of the above is not hard, but outside the scope of this course. In fact, CNOT is not unique here and can be replaced with any entangling gate.
In other words, Entanglement + local ops is universal

Problem with discretization:
We cannot implement every unitary with a finite set of gates

We want discretization for error correction (will be explained later)

If we set ourselves to a discrete set of angles and a finite set of gates, then we know we can only implement a countable number of unitaries.
We have an infinite number of unitaries over the complex field.

Solution:
Instead of implementing every possible unitary, we allow ourselves to approximate them.
We make the definition of universality less strict

If we reset ourselves to a discrete set of angles and a finite set of gates, then we know we can only implement a countable number of unitaries.
We have an infinite number of unitaries over the complex field.

(Approximate universality)

We now have a universal gate set, but it is not very satisfactory since it contains **infinitely many** gates. For **practical, programmable** quantum computers, it will be necessary to restrict the gate set to a **finite** set as in classical computing. Part of this is due to **error correction** which we will get to late in the course.

Uncountably infinite since there are infinite amount of complex numbers with limitless amount of precision. All of these numbers cannot be stored.

Unlike the classical case, **there is no finite gate set which can implement every unitary**. The reason is $\mathbb{C}$, and hence 2×2 unitaries, is **uncountably infinite**. Instead, we may ask whether there exists a finite gate set which can **approximate** every unitary matrix.

(Gate approximation error)

A unitary U **approximates** another unitary V with **error**

$$E(U,V) = \max_{|\psi\rangle} \|(U-V)|\psi\rangle\|$$

Infinity norm.

If U and V are diagonal, then (U - V)|psi>

An important property which you will prove in homework is that **approximation error is additive**, in that.

If we need to distribute the error epsilon among k gates, we can make every gate have epsilon/k probability of error.
Ask professor about the case where the error is not distributed evenly.
Why can we make the assumption that each gate has an equal error?

$$E(U_1 U_2, V_1 V_2) \leq E(U_1, V_1) + E(U_2, V_2)$$

Each quantum gate has its own error.

A practical consequence is that if **every gate** in a circuit is approximated to error $\frac{\varepsilon}{k}$ for a **k-gate** circuit, then the **total approximation error** is at most

We can either pick k large and/or epsilon small. Do we do both?
Is there one we prefer?
If we add more gates we increase the complexity, but these can be trivial gates.
We can reduce the error but increase the complexity.

(Approximate universality)

How can we get an approximate gate set?

A set of gates Γ is **approximately universal for quantum computing** if any n-qubit unitary U can be **approximated to arbitrary precision** $\epsilon > 0$ by a circuit V over Γ. That is,

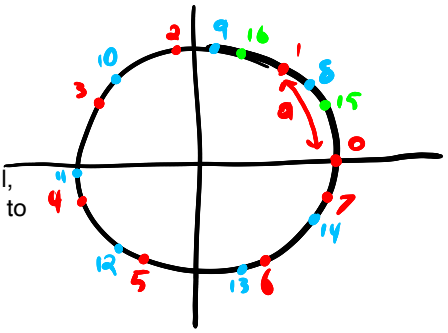
$$E(V, U) \leq \epsilon$$

Remember:
a unitary operation can be decomposed into three rotations on two non-paralell axes
We usually use the x and z axis, but it is more convenient to use other axes.

Since the {CNOT} U single-qubit gates set is **exactly** universal, it suffices to find a gate set which can approximate any single-qubit gate. The simplest way to do this is to write a single-qubit gate as

$$R_\ell(\alpha) R_k(\beta) R_\ell(\gamma)$$

for some non-parallel axes ℓ & k, then use $R_\ell(a), R_k(b)$ where a & b are **irrational multiples of π**. This means if we keep rotating around axis ℓ at an angle of a, we never get back to an earlier spot



If we have a rotation of alngle theta around axis l, then we can approximate (rotation-matrix-l alpha) to error e using (rotation-matrix-l theta) if theta != 2pi/k.

(Irrational rotations)

Recall that $T = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}$. The gates

Given H and T, we can construct irrational rotations.
We use these gates to approximate these rotations.

HTHT, THTH

are irrational rotations around non-parallel axes

(Corollary)

Approximate universality can be done using these three gates.

The set $\{CNOT, H, T\}$ is approximately universal.

called Clifford + T

(Efficiency of approximation)

Suppose we have an algorithm that runs on a quantum computer over the gate set CNOT + single-qubit unitaries in time poly(n), but to approximate the circuit over $\{CNOT, H, T\}$ we might use exp(n) gates! How can we be sure approximation will not incur exponential overhead? The seminal result of Solovay & Kitaev states that given certain assumptions (which have recently been proven unnecessary), approximations of single-qubit unitaries are efficient (polynomial).

Ask professor about the importance of the theorem

Before this theorem, people did not think that quantum computing was practical

(Solovay-Kitaev theorem)

Let Γ be a set of single qubit gates such that

1. Γ contains two rotations of angles which are irrational multiples of π around non-parallel axes, and
2. Γ is inverse-closed. That is, for every $U \in \Gamma$,

$$U^+ = U^{-1} \in \Gamma$$

Then any 1-qubit unitary may be approximated to error $\varepsilon > 0$ using $O(\log^c(1/\varepsilon))$ gates from Γ, where $c > 3$ for $\{H, T\}$.

Every time, we can take the commutator to change epsilon to 1/epsilon^{poly}

While the approximation factor for $\{H, T\}$ has been improved to $O(\log(1/\varepsilon))$ by other methods in recent years, the Solovay-Kitaev theorem was instrumental in showing that quantum computation was practically possible. In particular, without the Solovay-Kitaev theorem, error correction and hence general purpose quantum computers was a pipe-dream...

Midterm details:
Cheatsheet will be allowed (details to)
Will cover everything up to lecture 15 (most likely 14).
Will

Nielson and Chuang exercises are challanging (Take a look at it)
Use problem sets from Merman lectures

Physical implementation of single qubit unitaries:

Single qubit unitaries are not hard to implement. We only need to apply a magnetic field to a unitary in order to apply unitary. High fidelity

For multiple qubit unitaries, they are hard to implement. Low fidelity.
We can use the Ridbert Blockade.
If two particles which are in the Ridbert state (only one or othe other can be in the Ridbert state) are close together physically,

Implement CZ gate:
|psi> = |00> + |01> + |10> + |11>
-> |00> + |0r> + |10> + |1r>
Copying a state over n times does not entangle that state with any of the other EPR pairs used since you are independently copying it over to each state.

For reversible computing, you have m output wires which contains the bits of all your output states. You just do the CNOT of the wires which contain the results of applying the gates to the output bits and you copy the bits over, allowing you to have your result before wiping out the state of the qubits in the workspace and restoring the used qubits.

Suppose gate is an epsilon_0 net
e exists a gate which rotates one of these points to one that is epsilon_1 distance away.\