Quantum Computation (Preskill ch. 6)

Quantum Circuits

Classical Computer = finite set of gates acting on bits

Quantum Computer = finite set of quantum gates acting on quantum bits

Quantum Computation:

$U : |000\ldots0\rangle \rightarrow |\Psi\rangle$

unitary composed of finite # of gates

n qubit input

output = outcome of Orthog. Measurement in basis $\{|0\rangle, |1\rangle\}^n$

Note:

* The Hilbert space of the Quantum Computer has a preferred decomposition into tensor producs of low dimensional spaces (qubits), respected by gates which act on only a few qubits at a time.

  - This helps establish notion of Quantum Complexity

* Decomposition into subsystems and local manipulations means gates act on qubits in a bounded region.

* It is suspected, but not proven, that the power of Q. C. derives from this decomposition:

n qubits -> 2^n dimensional $\mathcal{H}$ resource grows $\sim 2^n$

* Unitaries form a continuum, but we restrict to discrete gate sets. This is necessary for Fault Tolerance

* Quantum Gates could be Superoperators, and readout could be POVM's

However:

we can simulate

Superoperators as unitaries
POVM's as Orthog. Meas

in larger $\mathcal{H}$

Our simpler conceptualization is general

Quantum Computation (Preskill ch. 6)

* It is suspected, but not proven, that the power of Q. C. derives from this decomposition:

n qubits -> 2^n dimensional $\mathcal{H}$ resource grows $\sim 2^n$

* Unitaries form a continuum, but we restrict to discrete gate sets. This is necessary for Fault Tolerance

* Quantum Gates could be Superoperators, and readout could be POVM's

However:

we can simulate | Superoperators as unitaries
POVM's as Orthog. Meas | in larger $\mathcal{H}$

Our simpler conceptualization is general

* Final readout could be collective or in a basis ≠ the standard logical basis →

  Unitary maps to standard basis $\{|0\rangle, |1\rangle\}^n$ with overhead included in complexity

* We could do measurements during computation, then condition later steps on the outcomes. But one can show the same results can be achieved by measuring at the end of the computation

  - In practice measurement during computation is essential for error correction

Note: None of the above changes notion of complexity

Quantum Computation (Preskill ch. 6)

* Final readout could be collective or in a basis $\neq$ the standard logical basis →

  Unitary maps to standard basis $\{|0\rangle, |1\rangle\}^n$ with overhead included in complexity

* We could do measurements during computation, then condition later steps on the outcomes. But one can show the same results can be achieved by measuring at the end of the computation

  - In practice measurement during computation is essential for error correction

Note: None of the above changes notion of complexity

At this point we are left with 3 main issues

(1) Universality: we must be able to implement the most general unitary $\in SU(2^n)$

group of unitaries in $\mathcal{H}$, $\mathrm{Dim}\,\mathcal{H} = 2^n$

→ Circuit of chosen gates must approx. any $U \in SU(2^n)$

(2) Quantum Complexity: New class BQP [1)]

Decision problems solved w/high prob. by poly-sized quantum circuits

(3) Accuracy: BQP is defined assuming perfect gates. What happens if circuit elements do not have exponential accuracy?

Can show noisy gates are OK: T - gate circuit requires error prob. $\propto 1/T$

[1)] BQP = Bounded-error Quantum Polynomial time

Quantum Computation (Preskill ch. 6)

At this point we are left with **3 main issues**

(1) Universality: we must be able to implement the most general unitary $\in SU(2^n)$

group of unitaries in $\mathcal{H}$, $\mathrm{Dim}\,\mathcal{H} = 2^n$

→ Circuit of chosen gates must approx. any $U \in SU(2^n)$

(2) Quantum Complexity: New class BQP[1]

Decision problems solved w/high prob. by poly-sized quantum circuits

(3) Accuracy: BQP is defined assuming perfect gates. What happens if circuit elements do not have exponential accuracy?

Can show noisy gates are OK: T - gate circuit requires error prob. $\propto 1/T$

[1] BQP = Bounded-error Quantum Polynomial time

Note on Quantum Complexity:

A QC can simulate a probabilistic classical computer (most general class)

→ $\boxed{BPP^{2)} \subseteq BQP}$

Open Question: Is $BPP \neq BQP$? Seems reasonable, as a prob. C.C. cannot easily simulate QM in a 2^n – dimensional Hilbert space.

If so, a QC will negate the Strong Church-Turing Thesis which holds that any physically reasonable model of computation can be simulated on a probabilistic classical computer with only polynomial slowdown.

[1] BQP = Bounded-error Quantum Polynomial time

[2] BPP = Bounded-error Probabilistic Polynomial time

Quantum Computation (Preskill ch. 6)

Universal Quantum Gates

* What constitutes a universal gate set ?

  Answer: Almost any generic 2–qubit quantum gate will do!

* What is a generic gate?

A k–qubit gate $U = 2^k \times 2^k$ matrix w/evals $\{e^{i\theta_1}, \dots e^{i\theta_{2^k}}\}$ is **generic** if

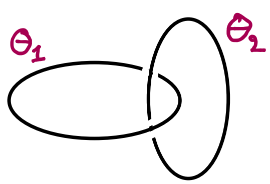
θ_i is an irrational multiple of π

θ_i, θ_j are incommensurate (θ_i/θ_j irrational multiple of π)

(1) **Powers of a generic gate**:

$U^n \rightarrow$ evals $\{e^{in\theta_1}, \dots, e^{in\theta_{2^k}}\}$

points on 2^k dim torus



$\left.\begin{array}{l} U \text{ generic} \\ n \in N_0 \end{array}\right\} \Rightarrow$ points densely covers the whole torus

Definition:

Let $U = e^{iH_j\,dt}$ be **generic** (H_j is the generator of U)

$\exists n \in N_0$ so U^n comes **arbitrarily close** to $U(\alpha) = e^{i\alpha H_j}$

($U(\alpha)$ is **reachable** by powers U^n)

Seems extraordinarily cumbersome! Why do it that way?

Answer: This is necessary for **Fault Tolerant Operation**

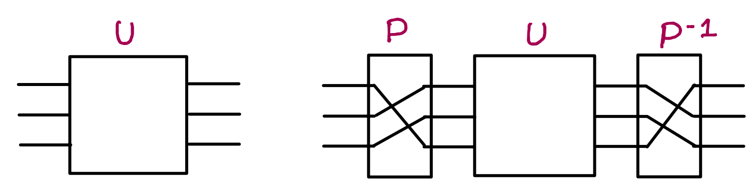
$\{U^n, n \in N_0\}$ is a set of measure zero $\rightarrow$ any "noise takes us to an invalid state that can be detected and corrected.

This is not enough! What else can we do?

(2) **Switching leads**

k qubits $\rightarrow$ $(2^k)!$ **permutations** $U' = P\,U\,P^{-1}$

$k = 3$



Quantum Computation (Preskill ch. 6)

Definition:

Let $U = e^{iH_j dt}$ be generic (H_j is the generator of U)

$\exists\, n \in \mathbb{N}_0$ so U^n comes arbitrarily close to $U(\alpha) = e^{i\alpha H_j}$

($U(\alpha)$ is reachable by powers U^n)

Seems extraordinarily cumbersome! Why do it that way?

Answer: This is necessary for Fault Tolerant Operation

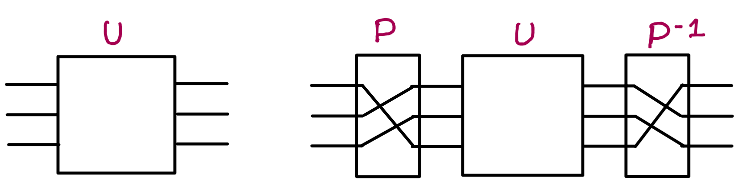
$\{U^n, n \in \mathbb{N}_0\}$ is a set of measure zero → any "noise takes us to an invalid state that can be detected and corrected.

This is not enough! What else can we do?

(2) Switching leads

k qubits → $(2^k)!$ permutations $U' = P\,U\,P^{-1}$

k = 3



This is not enough! What else can we do?

Aside: Consider a d-dimensional Hilbert space $\mathcal{H}$.

⇨ Operators ($d \times d$ matrices) are vectors $\in d^2$ dim. Hilbert space $\mathcal{H}'$ w/a scalar product defined as

$$(m_i | m_j) = \mathrm{Tr}[m_i^\dagger m_j]$$

$\exists$ orthonormal basis $\left\{ \begin{array}{l} \{|A_1), \dots |A_{d^2})\} \\ (A_i | A_j) = \delta_{ij} \end{array} \right.$ in $\mathcal{H}'$

(3) Completing the Lie Algebra

Assume access to a set of Hamiltonians

$$\{H_0, H_1, \dots H_n\},\ n \leq (\dim \mathcal{H})^2$$

Trotter Formulae:

$$e^{-i\alpha H_j dt} e^{-i\beta H_k dt} = e^{-i(\alpha H_j + \beta H_k)dt}$$

$$e^{-i\alpha H_j dt} e^{-i\beta H_k dt} e^{i\alpha H_j dt} e^{i\beta H_k dt} = e^{-[\alpha H_j, \beta H_k] dt^2}$$

Quantum Computation (Preskill ch. 6)

Aside: Consider a d - dimensional Hilbert space $\mathcal{H}$.

Operators ($d \times d$ matrices) are vectors $\in d^2$ dim. vector space $\mathcal{H}'$ w/a scalar product defined as

$$(m_i | m_j) = \mathrm{Tr}[m_i^{+} m_i]$$

$\exists$ orthonormal basis $\begin{cases} \{|A_1),\dots |A_{d^2})\} \\ (A_i | A_j) = \delta_{ij} \end{cases}$ in $\mathcal{H}'$

(3) Completing the Lie Algebra

Assume access to a set of Hamiltonians

$$\{H_0, H_1, \dots H_n\},\ n \leq (\dim \mathcal{H})^2$$

Trotter formulae:

$$e^{-i\alpha H_j dt} e^{-i\beta H_k dt} = e^{-i(\alpha H_j + \beta H_k)dt}$$

$$e^{-i\alpha H_j dt} e^{-i\beta H_k dt} e^{i\alpha H_j dt} e^{i\beta H_k dt} = e^{-[\alpha H_j, \beta H_k] dt^2}$$

* From the Set $\{H_0, H_1, \dots H_n\}$ we can "simulate" new Hamiltonians using the Trotter formulae
* If a new Hamiltonian is linearly independent we add it to the set.
* Continue until the Set has $d^2 = (\dim \mathcal{H})^2$ linearly independent members (Lie Algebra complete)*)

Set is a basis in $d^2 \times d^2$ matrix space

Allows to simulate any $H(t)$ & implement any U

Examples:

$d = 2 \rightarrow \{|A_i)\} = \{I, \sigma_x, \sigma_y, \sigma_z\} \leftarrow$ set of $2^2 = 4$ 2 x 2 matrices

$d = 4 \rightarrow \{|A_i)\} \leftarrow$ set of $d^2 = 16$ 4 x 4 matrices

Example: (single qubit control)

Let $d = 2$, initial set $\{\alpha\sigma_x, \beta\sigma_y\}$ (generic)

$[\sigma_x, \sigma_y] = i\sigma_z \rightarrow$ we can simulate $i\gamma\sigma_z$

*) This is not always possible. The Lie Algebra may "close" before generating a basis. If so, add more Hamiltonians to the original set.

Quantum Computation (Preskill ch. 6)

Aside: Consider a d - dimensional Hilbert space $\mathcal{H}$.

$\begin{cases} \text{Operators } (d \times d \text{ matrices}) \text{ are vectors} \in d^2 \text{ dim.} \\ \text{vector space } \mathcal{H}' \text{ w/a scalar product defined as} \end{cases}$

$$(m_i | m_j) = \mathrm{Tr}[m_i^+ m_i]$$

$\exists$ orthonormal basis $\begin{cases} \{ |A_1), \dots |A_{d^2}) \} \\ (A_i | A_j) = \delta_{ij} \end{cases}$ in $\mathcal{H}'$

(3) Completing the Lie Algebra

Assume access to a set of Hamiltonians

$$\{H_0, H_1, \dots H_n\}, \ n \leq (\dim \mathcal{H})^2$$

Trotter formulae:

$$e^{-i\alpha H_j dt} e^{-i\beta H_k dt} = e^{-i(\alpha H_j + \beta H_k) dt}$$

$$e^{-i\alpha H_j dt} e^{-i\beta H_k dt} e^{i\alpha H_j dt} e^{i\beta H_k dt} = e^{-[\alpha H_j, \beta H_k] dt^2}$$

* From the Set $\{H_0, H_1, \dots H_n\}$ we can "simulate" new Hamiltonians using the Trotter formulae
* If a new Hamiltonian is linearly independent we add it to the set.
* Continue until the Set has $d^2 = (\dim \mathcal{H})^2$ linearly independent members (Lie Algebra complete) *)

Set is a basis in $d^2 \times d^2$ matrix space

Allows to simulate any $H(t)$ & implement any U

Examples:

$d = 2 \rightarrow \{ |A_i) \} = \{ I, \sigma_x, \sigma_y, \sigma_z \} \leftarrow$ set of $2^2 = 4$ 2 x 2 matrices

$d = 4 \rightarrow \{ |A_i) \} \leftarrow$ set of $d^2 = 16$ 4 x 4 matrices

Example: (single qubit control)

Let $d = 2$, initial set $\{ \alpha\sigma_x, \beta\sigma_y \}$ (generic)

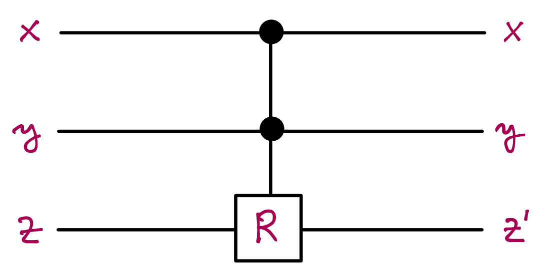
$[\sigma_x, \sigma_y] = i\sigma_z \rightarrow$ we can simulate $i\gamma\sigma_z$

Set $[I, \alpha\sigma_x, \beta\sigma_y]$ sufficient for control

Quantum Computation (Preskill ch. 6)

Deutsch's Gate

(First generic gate, Reaches any $U \in SU(8)$)



Rotation $R = -iR_x(\theta) = -ie^{i\theta/2\,\sigma_x} = -i\left(\cos\frac{\theta}{2} + i\sigma_x \sin\frac{\theta}{2}\right)$ iff $xy = 1$

incommensurate w/π

Special case $\theta = \pi$ ➡ this is a Toffoli gate: $-iR_x(\pi) = -i\sigma_x$

to within a phase

Note: $R^{4n} = R_x(4n\theta)$ (b/c $i^4 = 1$) ➡

$$R^{(4n+1)} = (-i)\left[\cos\frac{(4n+1)\theta}{2} + i\sigma_x \sin\frac{(4n+1)\theta}{2}\right] \simeq \sigma_x \text{ for some } n$$

Action on the basis states: $R^{(4n+1)}$ transposes (6) & (7)

$$\{\underset{(0)}{|000\rangle}, \underset{(1)}{|001\rangle}, \underset{(2)}{|010\rangle}, \underset{(3)}{|011\rangle}, \underset{(4)}{|100\rangle}, \underset{(5)}{|101\rangle}, \underset{(6)}{|110\rangle}, \underset{(7)}{|111\rangle}\} \quad *)$$

$$|111\rangle, |110\rangle$$

σ_x on 2D subspace

Note: A Deutsch gate on a 3-qubit state can be cast as an 8 x 8 matrix acting in an 8-dimensional vector space.

With the basis states numbered as in $*$) above, $R^{(4n+1)}$ has the matrix representation

$$(\sigma_x)_{67} = \left(\begin{array}{c|c} 0 & 0 \\ \hline 0 & \sigma_x \end{array}\right)$$ ⬅ flips the spin of the 2-level system (6),(7)

By switching leads and applying Toffoli gates, we can do any perturbation of basis states. Thus we can reach

$$\boxed{P(\sigma_x)_{67}P^{-1} = (\sigma_x)_{nm}}$$

In turn, this allows us to reach $e^{i(\sigma_x)_{56}}$ and $e^{i(\sigma_x)_{67}}$

➡ we can reach $e^{-[(\sigma_x)_{56}, (\sigma_x)_{67}]}$

On matrix form: $$[(\sigma_x)_{56}, (\sigma_x)_{67}] = \left[\left(\begin{array}{c|c} 0 & 0 \\ \hline 0 & \begin{smallmatrix} 0&1&0\\1&0&0\\0&0&0 \end{smallmatrix} \end{array}\right), \left(\begin{array}{c|c} 0 & 0 \\ \hline 0 & \begin{smallmatrix} 0&0&0\\0&0&1\\0&1&0 \end{smallmatrix} \end{array}\right)\right] = (\sigma_y)_{57}$$

Thus: $\boxed{\text{Compositions of } (\sigma_x)_{nm}\text{'s} \rightarrow i(\sigma_y)_{pq}\text{'s}}$

these generators   these generators

Quantum Computation (Preskill ch. 6)

Note: A Deutsch gate on a 3-qubit state can be cast as an 8 x 8 matrix acting in an 8-dimensional vector space.

With the basis states numbered as in $*)$ above, $R^{(4n+1)}$ has the matrix representation

$$(\sigma_x)_{67} = \left(\begin{array}{c|c} 0 & 0 \\ \hline 0 & \sigma_x \end{array}\right) \longleftarrow$$ flips the spin of the 2-level system (6),(7)

By switching leads and applying Toffoli gates, we can do any perturbation of basis states. Thus we can reach

$$\boxed{P(\sigma_x)_{67}P^{-1} = (\sigma_x)_{nm}}$$

In turn, this allows us to reach $e^{i(\sigma_x)_{56}}$ and $e^{i(\sigma_x)_{67}}$

we can reach $e^{-[(\sigma_x)_{56},(\sigma_x)_{67}]}$

On matrix form: $$[(\sigma_x)_{56},(\sigma_x)_{67}] = \left[\left(\begin{array}{c|c} 0 & 0 \\ \hline 0 & \begin{matrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{matrix} \end{array}\right), \left(\begin{array}{c|c} 0 & 0 \\ \hline 0 & \begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{matrix} \end{array}\right)\right] = (\sigma_y)_{57}$$

Thus: Compositions of $(\sigma_x)_{nm}$'s $\rightarrow$ $i(\sigma_y)_{pq}$'s

these generators these generators

And thus we can make up

$e^{i\frac{\theta}{2}(\sigma_y)_{pq}}$'s from powers of the $e^{i\frac{\varphi}{2}(\sigma_x)_{nm}}$'s,

which in turn can be obtained from powers of $-iR_x(\gamma)$

generic Deutsch gate

Similarly, $[(\sigma_x)_{nm},(\sigma_y)_{nm}] = i(\sigma_z)_{nm}$ $\Rightarrow$

Compositions of $(\sigma_x)_{nm}$'s & $(\sigma_y)_{nm}$'s $\rightarrow$ $(\sigma_z)_{nm}$'s

Conclusion: We can reach all transformations generated by linear combinations of the $(\sigma_{x,y,z})_{nm}$'s, which together span the SU(8) Lie Algebra

Quantum Computation (Preskill ch. 6)

And thus we can make up

$e^{i\frac{\theta}{2}(\sigma_y)_{pq}}$'s from powers of the $e^{i\frac{\varphi}{2}(\sigma_x)_{nm}}$'s,

which in turn can be obtained from powers of $-iR_x(\gamma)$

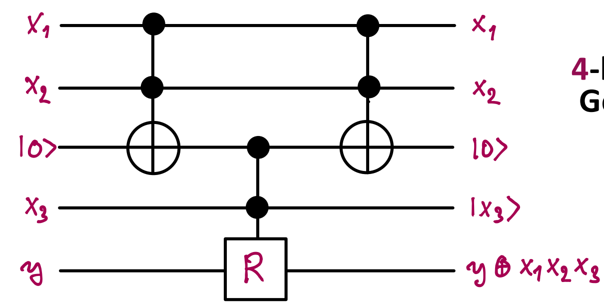
generic Deutsch gate

Similarly, $[(\sigma_x)_{nm}, (\sigma_y)_{nm}] = i(\sigma_z)_{nm}$

Compositions of $(\sigma_x)_{nm}$'s & $(\sigma_y)_{nm}$'s $\rightarrow$ $(\sigma_z)_{nm}$'s

Conclusion: We can reach all transformations generated by linear combinations of the $(\sigma_{x,y,z})_{nm}$'s, which together span the SU(8) Lie Algebra

Extending to *n* bit Deutsch gate:



4-bit Deutsch Gate Generates SU(2^4)

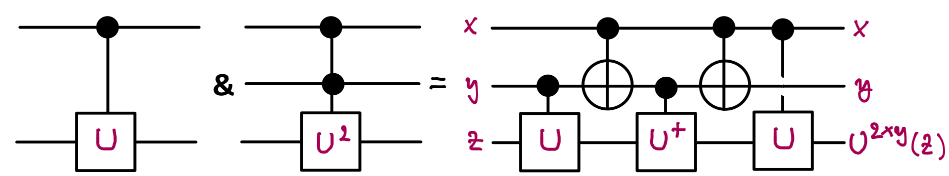
Repeat $\rightarrow$ *n* bit Deutsch gate generates SU(2^n)

The Deutsch Gate is Universal

Universal 2-qubit gate sets

Proof: can build a Deutsch gate from 2-qubit gates

Assume we have | Then **we can do**



We can build a Deutsch gate from

- controlled U
- controlled U^{-1}
- controlled NOT