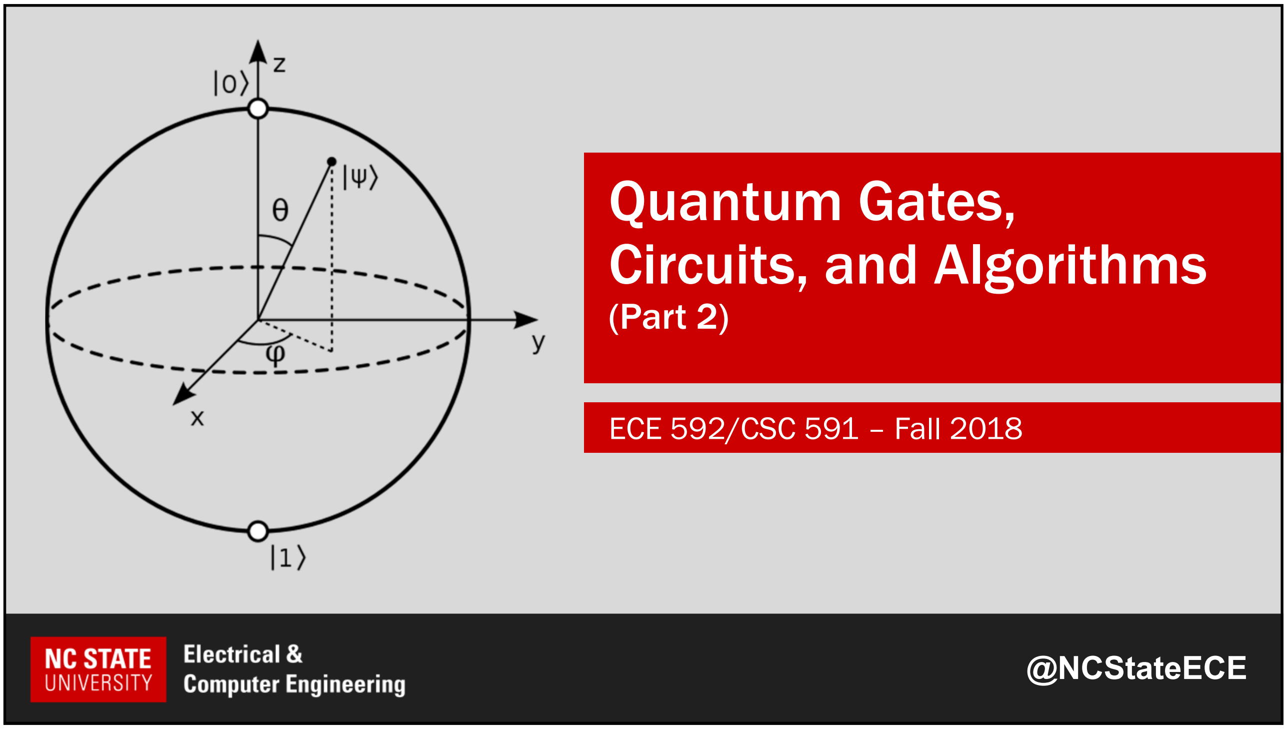


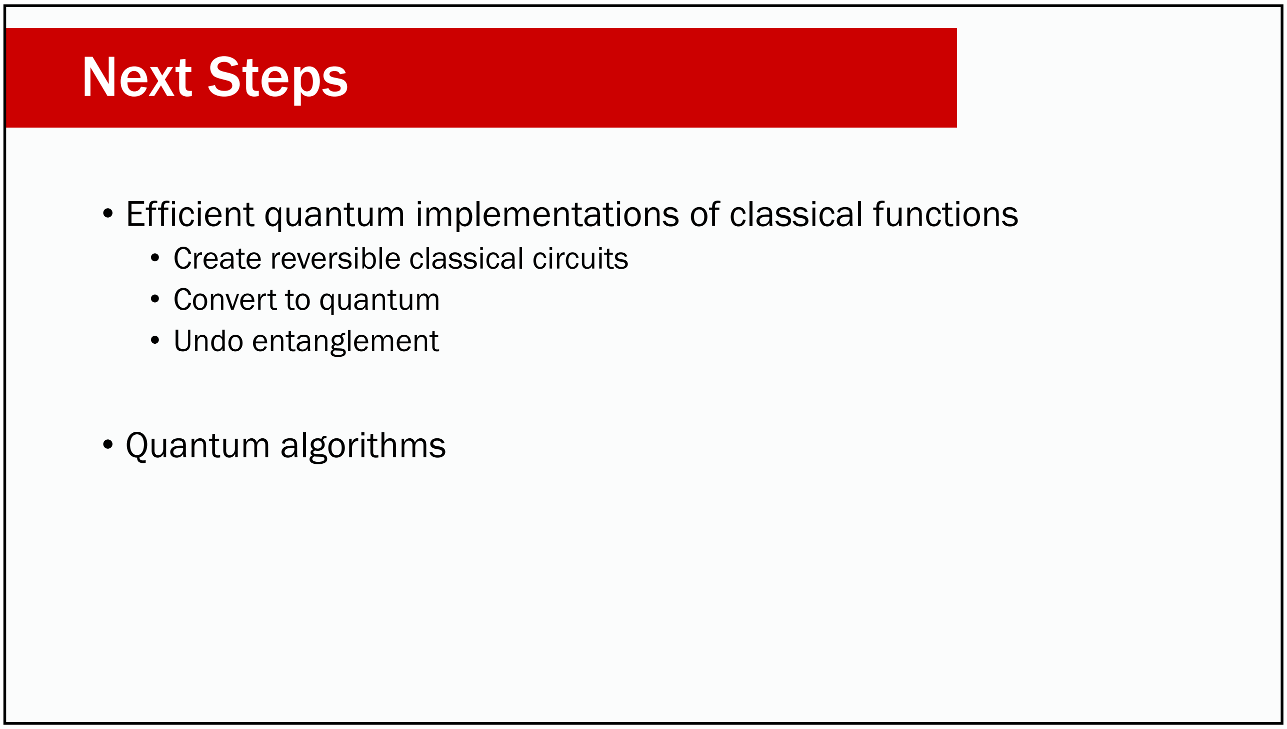
Quantum Gates, Circuits, and Algorithms

(Part 2)

ECE 592/CSC 591 – Fall 2018

NC STATE UNIVERSITY Electrical & Computer Engineering

@NCStateECE

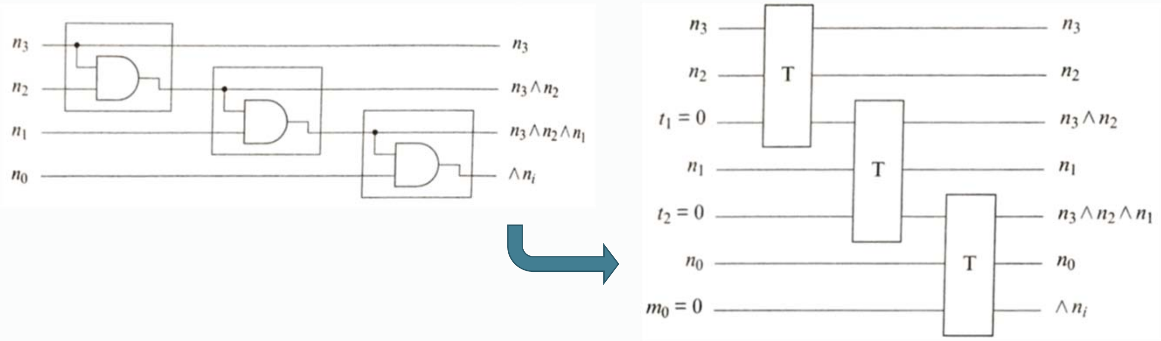


Next Steps

- Efficient quantum implementations of classical functions
    - Create reversible classical circuits
    - Convert to quantum
    - Undo entanglement
- Quantum algorithms

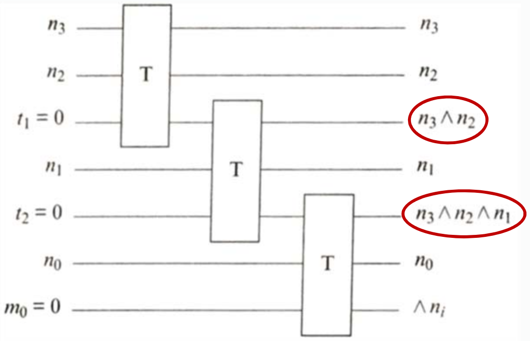
Classical (Boolean) Circuits

In general, classical circuits are not reversible. Consider this circuit for a 4-way AND.

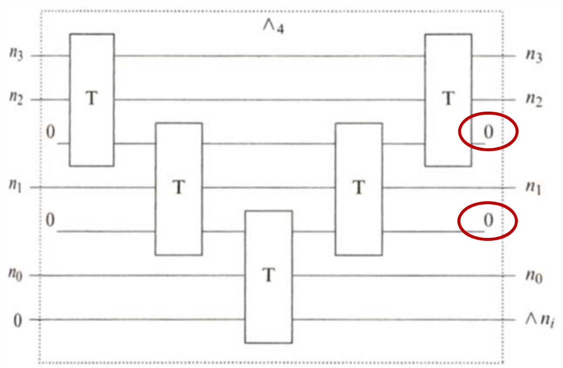


By using (classical) Toffoli gate, we can construct a reversible circuit. Requires one additional bit per AND gate, and of course a Toffoli gate is more complex than an AND gate.

Reusing Temporary Bits



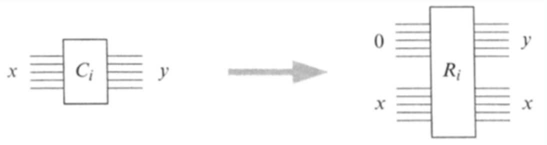
These bits are no longer zero, and can't be reused if this feeds into additional computation. Can't just "reset" them, because that's not reversible. Need to uncompute to reclaim them.

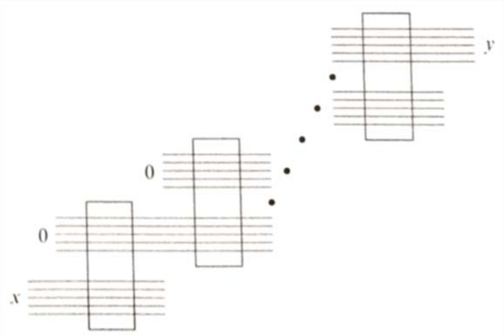


Tradeoff: Uncomputing requires extra gates, so when is it better to reclaim vs. retain for potential later use?

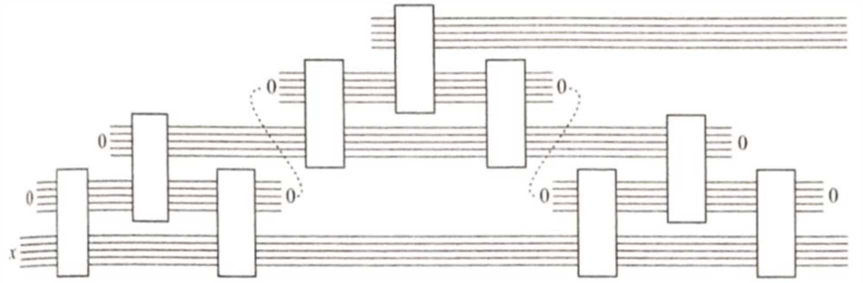
General Scheme

Assume a classical circuit C can be decomposed into subcircuits C_i, convert each to reversible R_i.





General Scheme



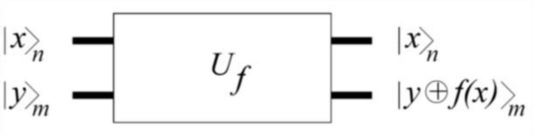
Using this general scheme, any classical circuit with t steps and s bits can be done reversibly in $O(t^{1+\varepsilon})$ steps and $O(s \log t)$ bits. There may be more efficient implementations for a specific circuit.

Uncomputing temporary qubits is more important in the quantum realm, because:

- Qubits are more precious than bits. Reducing storage is more important (for now) than reducing gates.
- Temporary bits may be entangled with output bits.
  Measuring them to reset, for use in later computation, can disturb the output bits.

Summary

- Any computation with an efficient classical circuit has an equivalent efficient quantum circuit.



- Use reverse computation to unentangle and reuse tmp qubits.

Further reading:

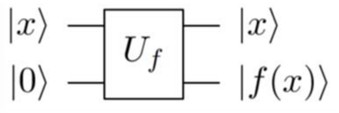
- John Preskill notes, Chapter 6
- Vedral, Barenco, and Ekert, Quantum Networks for Elementary Arithmetic Operations, *Physical Review A*, 54(1):147-153, 1996.
- Barenco, et al., Elementary Gates for Quantum Computation, *Physical Review A*, 52(5):3457-3467, 1995.

Simple Quantum Algorithms

- Deutsch
- Phase Change for a Subset of Basis Vectors
- Deutsch-Josza
- Simon

Deutsch Algorithm

Problem: Given a Boolean function $f\colon \mathbb{Z}_2 \to \mathbb{Z}_2$, determine whether f is constant.



Apply U_f to the input state $|+\rangle|-\rangle$.

If $f(x)$ is constant, then output is $|+\rangle|-\rangle$.
If not, output is $|-\rangle|-\rangle$.

Apply Hadamard to first qubit and measure: 1 if constant, 0 if not.

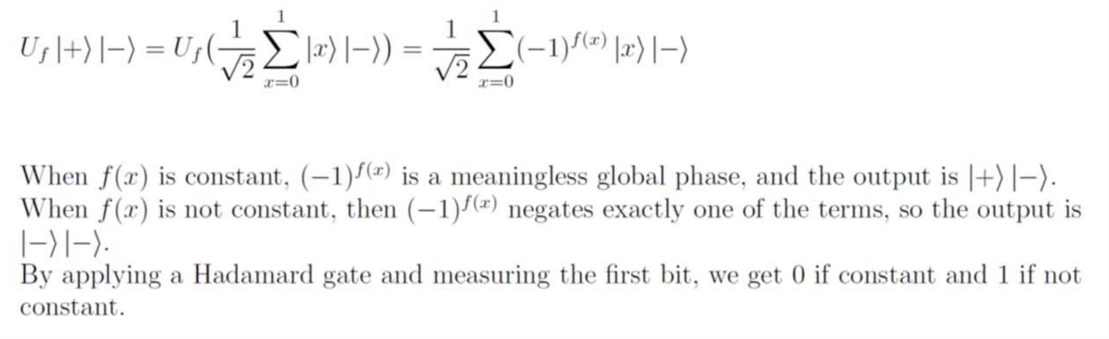
(Details on next slides.)

Requires only a single call to black box U_f, while classical algorithm requires two calls.

$$
\begin{aligned}
U_f\,|+\rangle\,|-\rangle &= U_f\big(\frac{1}{2}(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)\big) \\
&= \frac{1}{2}\big(|0\rangle\,(|0 \oplus f(0)\rangle - |1 \oplus f(0)\rangle) + |1\rangle\,(|0 \oplus f(1)\rangle - |1 \oplus f(1)\rangle)\big) \\
&= \frac{1}{2}\sum_{x=0}^{1} |x\rangle\,(|0 \oplus f(x)\rangle - |1 \oplus f(x)\rangle)
\end{aligned}
$$

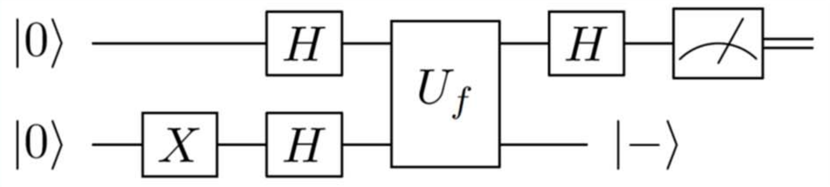
When $f(x) = 0$, this becomes $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = |-\rangle$.
When $f(x) = 1$, this becomes $\frac{1}{\sqrt{2}}(|1\rangle - |0\rangle) = -\,|-\rangle$.

$$U_f\,|+\rangle\,|-\rangle = U_f\Big(\frac{1}{\sqrt{2}}\sum_{x=0}^{1} |x\rangle\,|-\rangle\Big) = \frac{1}{\sqrt{2}}\sum_{x=0}^{1}(-1)^{f(x)}\,|x\rangle\,|-\rangle$$



$$U_f\left|+\right\rangle\left|-\right\rangle = U_f\Big(\frac{1}{\sqrt{2}}\sum_{x=0}^{1}\left|x\right\rangle\left|-\right\rangle\Big) = \frac{1}{\sqrt{2}}\sum_{x=0}^{1}(-1)^{f(x)}\left|x\right\rangle\left|-\right\rangle$$

When $f(x)$ is constant, $(-1)^{f(x)}$ is a meaningless global phase, and the output is $\left|+\right\rangle\left|-\right\rangle$. When $f(x)$ is not constant, then $(-1)^{f(x)}$ negates exactly one of the terms, so the output is $\left|-\right\rangle\left|-\right\rangle$.
By applying a Hadamard gate and measuring the first bit, we get 0 if constant and 1 if not constant.



Selective Phase Change

Problem: Change the phase of terms in a superposition $|\psi\rangle = \sum a_i\,|i\rangle$, depending on whether i is in a subset X of $\{0,1,\ldots,N-1\}$ or not. More specifically, find an efficient implementation of the following transform:

$$S_X^{\phi}: \sum_{x=0}^{N-1} a_x|x\rangle \rightarrow \sum_{x\in X} a_x e^{i\phi}|x\rangle + \sum_{x\notin X} a_x|x\rangle$$

Requires an efficient implementation of U_f for the function $f(x)$ that tests for membership in X:

$$f(x) = \begin{cases} 1 & \text{if } x \in X \\ 0 & \text{otherwise} \end{cases}$$

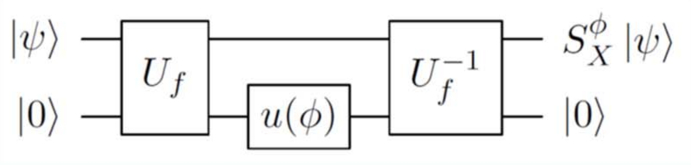
First, apply U_f to $|\psi\rangle\,|0\rangle$.

$$\begin{aligned}|\gamma\rangle = U_f\,|\psi\rangle\,|0\rangle &= \sum_x a_x\,|x, f(x)\rangle \\ &= \sum_{x\in X} a_x\,|x\rangle\,|1\rangle + \sum_{x\notin X} a_x\,|x\rangle\,|0\rangle\end{aligned}$$

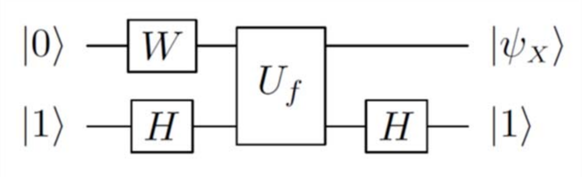
Now, apply the phase change gate $u(\phi) = \left(\begin{smallmatrix}1 & 0\\ 0 & e^{i\phi}\end{smallmatrix}\right)$ to the output qubit. This has no effect on $|0\rangle$ and shifts the phase on $|1\rangle$.

$$\begin{aligned}(I^{\otimes n}\otimes u(\phi))\,|\gamma\rangle &= \sum_{x\in X} a_x\,|x\rangle\,e^{i\phi}\,|1\rangle + \sum_{x\notin X} a_x\,|x\rangle\,|0\rangle \\ &= \sum_{x\in X} a_x e^{i\phi}\,|x\rangle\,|1\rangle + \sum_{x\notin X} a_x\,|x\rangle\,|0\rangle \\ &= \left(\sum_{x\in X} a_x e^{i\phi}\,|x\rangle + \sum_{x\notin X} a_x\,|x\rangle\right)|f(x)\rangle \\ &= (S_X^\phi\,|\psi\rangle)\,|f(x)\rangle\end{aligned}$$

Finally, uncompute using $U_f^{-1}$to remove any entanglement with the output bit.



Special case of π:



$$|\psi_X\rangle = \frac{1}{\sqrt{N}}\sum_x (-1)^{f(x)}\,|x\rangle$$

Background: Hamming Distance

The Hamming distance $\boldsymbol{d_H(x,y)}$ between two bit strings x and y is the number of bits in which the two strings differ.

The Hamming weight $\boldsymbol{d_H(x)}$ of a bit string x is the number of 1 bits.

For two bit strings x and y, the operator $\boldsymbol{x \cdot y}$ gives the number of common 1 bits.

Some interesting notes:

$$x \cdot y = d_H(x \wedge y)$$

$$\sum_{x=0}^{2^n-1} (-1)^{x \cdot x} = 0$$

$$\sum_{x=0}^{2^n-1} (-1)^{x \cdot y} = \begin{cases} 2^n & \text{if } y = 0 \\ 0 & \text{otherwise} \end{cases}$$

More on Walsh-Hadamard

$$W\left|0\right\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} \left|x\right\rangle$$

$$\begin{aligned} W\left|r\right\rangle &= (H \otimes \cdots \otimes H)(\left|r_{n-1}\right\rangle \otimes \cdots \otimes \left|r_0\right\rangle) \\ &= \frac{1}{\sqrt{2^n}}(\left|0\right\rangle + (-1)^{r_{n-1}}\left|1\right\rangle) \otimes \cdots \otimes (\left|0\right\rangle + (-1)^{r_0}\left|1\right\rangle) \\ &= \frac{1}{\sqrt{2^n}} \sum_{s=0}^{2^n-1} (-1)^{s_{n-1}r_{n-1}}\left|s_{n-1}\right\rangle \otimes \cdots \otimes (-1)^{s_0 r_0}\left|s_0\right\rangle \\ &= \frac{1}{\sqrt{2^n}} \sum_{s=0}^{2^n-1} (-1)^{s \cdot r}\left|s\right\rangle \end{aligned}$$

Deutsch-Josza Algorithm

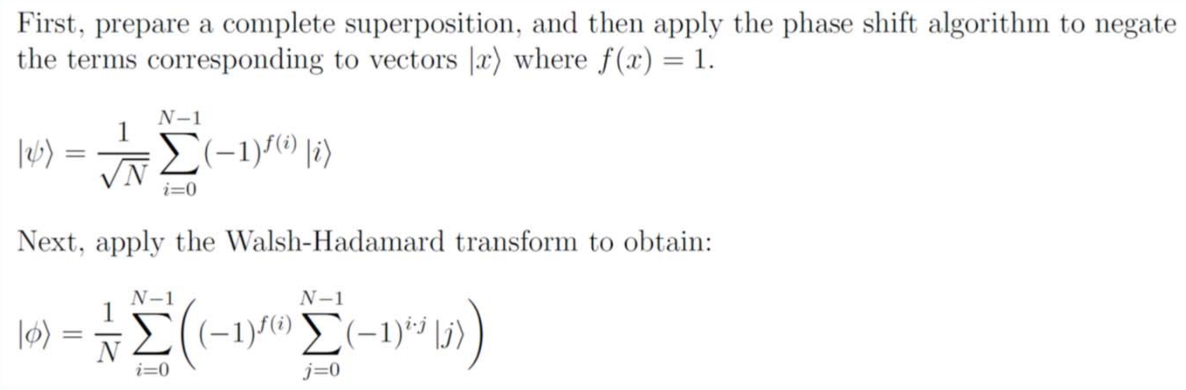
Problem: Given an *n*-bit Boolean function (mapping *n* bits to 1) that is known to be either constant or balanced, determine whether it is balanced or constant. A function is “balanced” if an equal number of input values return 0 and 1.

Apply phase shift of π to negate elements where $f(x) = 1$.
Apply Walsh-Hadamard to the result.

For constant f, the final output is $|0\rangle$ with probability 1.
For balanced f, the final output is non-zero with probability 1.

(Details on next slides.)

Requires only a single call to black box U_f, while classical algorithm requires at least $2^{n-1} + 1$ calls.

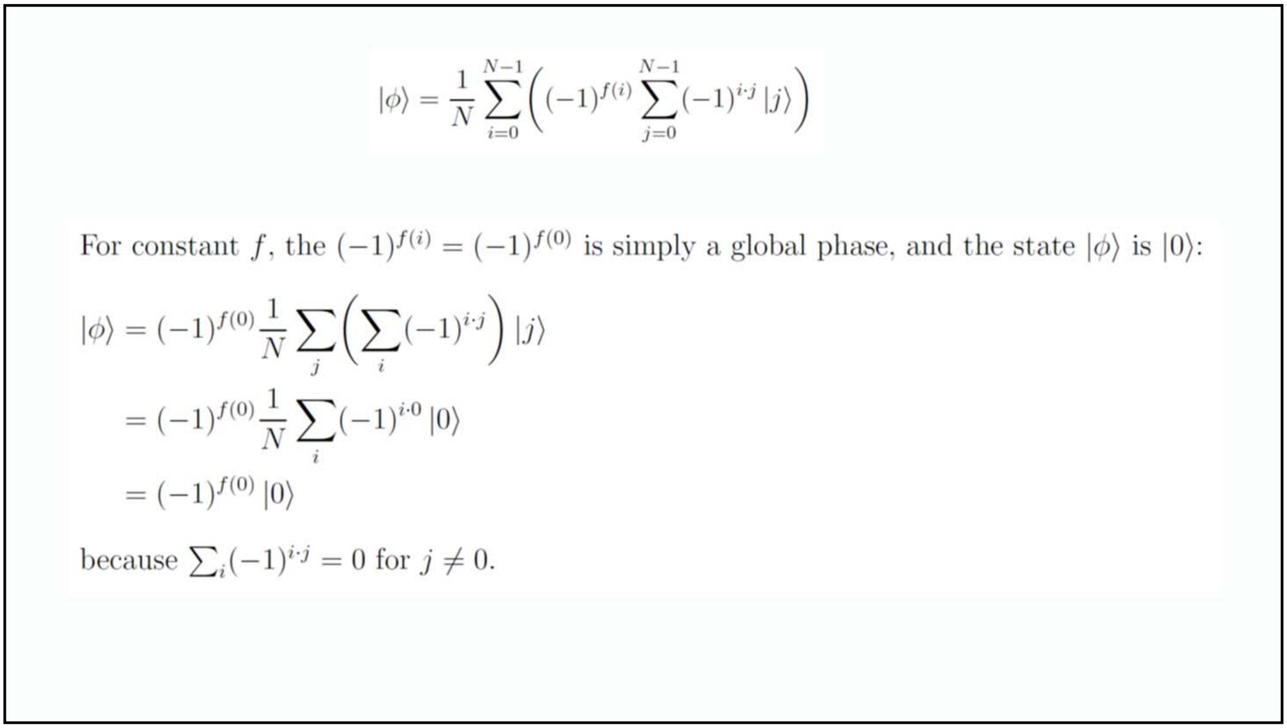


First, prepare a complete superposition, and then apply the phase shift algorithm to negate the terms corresponding to vectors $|x\rangle$ where $f(x) = 1$.

$$|\psi\rangle = \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} (-1)^{f(i)} |i\rangle$$

Next, apply the Walsh-Hadamard transform to obtain:

$$|\phi\rangle = \frac{1}{N} \sum_{i=0}^{N-1} \left((-1)^{f(i)} \sum_{j=0}^{N-1} (-1)^{i \cdot j} |j\rangle \right)$$

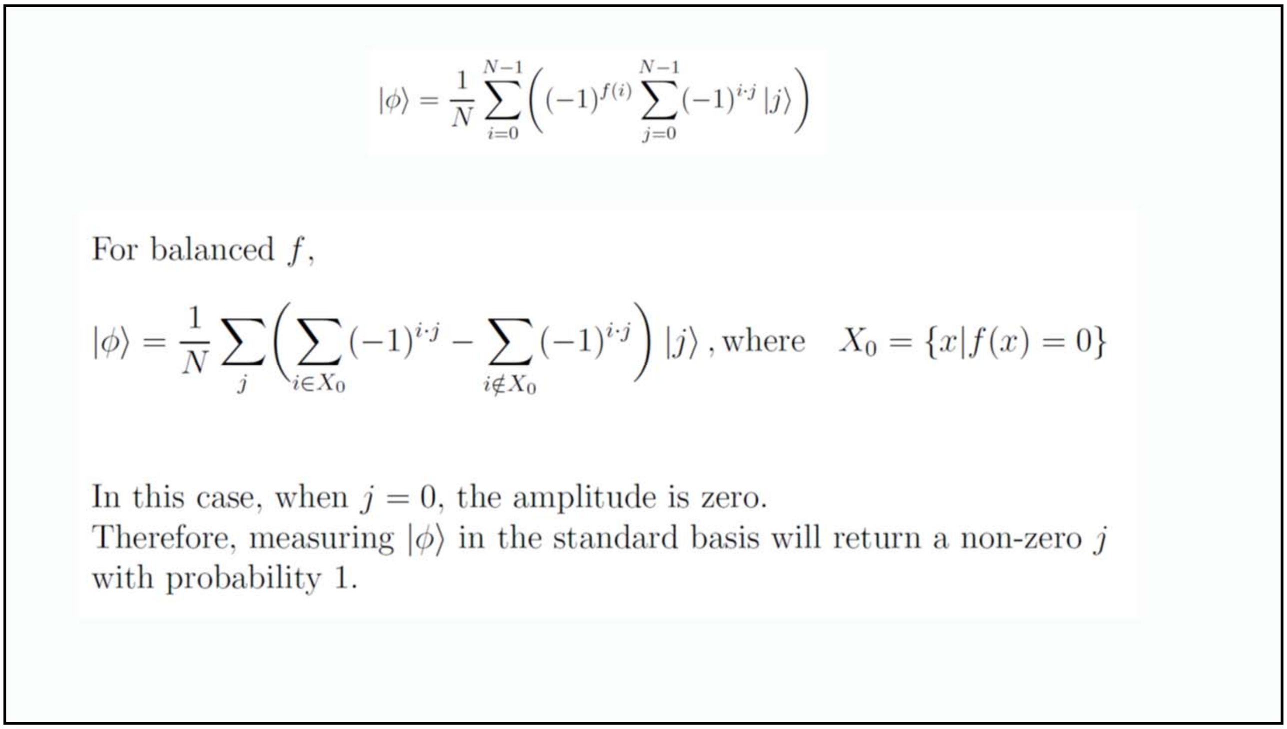


$$|\phi\rangle = \frac{1}{N}\sum_{i=0}^{N-1}\left((-1)^{f(i)}\sum_{j=0}^{N-1}(-1)^{i\cdot j}\,|j\rangle\right)$$

For constant f, the $(-1)^{f(i)} = (-1)^{f(0)}$ is simply a global phase, and the state $|\phi\rangle$ is $|0\rangle$:

$$\begin{aligned}|\phi\rangle &= (-1)^{f(0)}\frac{1}{N}\sum_j\left(\sum_i(-1)^{i\cdot j}\right)|j\rangle\\ &= (-1)^{f(0)}\frac{1}{N}\sum_i(-1)^{i\cdot 0}\,|0\rangle\\ &= (-1)^{f(0)}\,|0\rangle\end{aligned}$$

because $\sum_i(-1)^{i\cdot j} = 0$ for $j \neq 0$.



$$|\phi\rangle = \frac{1}{N}\sum_{i=0}^{N-1}\left((-1)^{f(i)}\sum_{j=0}^{N-1}(-1)^{i\cdot j}\,|j\rangle\right)$$

For balanced f,

$$|\phi\rangle = \frac{1}{N}\sum_j\left(\sum_{i\in X_0}(-1)^{i\cdot j} - \sum_{i\notin X_0}(-1)^{i\cdot j}\right)|j\rangle\,,\text{where}\quad X_0 = \{x|f(x)=0\}$$

In this case, when $j = 0$, the amplitude is zero.
Therefore, measuring $|\phi\rangle$ in the standard basis will return a non-zero j with probability 1.

Simon’s Algorithm

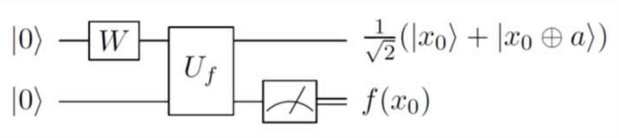
Problem: Given a 2-to-1 function f, such that $f(x) = f(x \oplus a)$, find the hidden string a.

Create superposition $|x\rangle|f(x)\rangle$
Measure the right part, which projects the left state into $\frac{1}{\sqrt{2}}(|x_0\rangle + |x_0 \oplus a\rangle)$.
Apply Walsh-Hadmard. (Details next slide.)

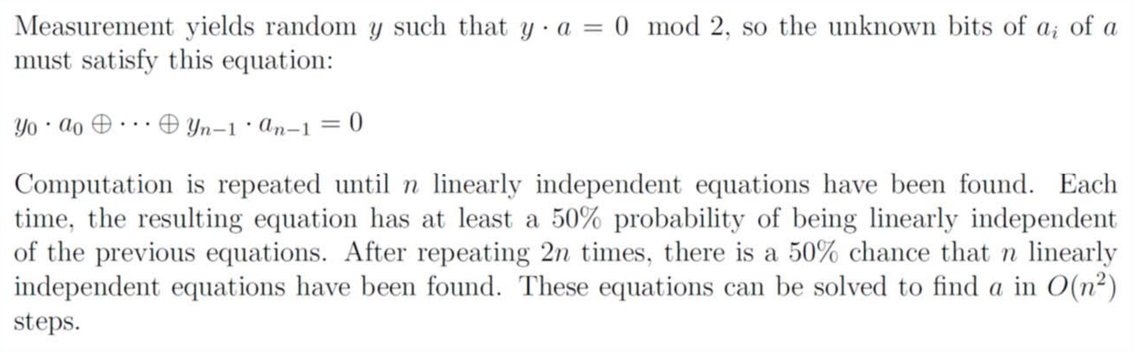
Measurement yields a random y such that $y \cdot a = 0$ (mod 2).
Computation is repeated until n independent equations – about $2n$ times.
Solve for a in $O(n^2)$ steps.

Requires $O(n)$ calls to U_f, followed by $O(n^2)$ steps to solve for a.

Classical approach requires $O(2^{n/2})$ calls to f.



$$\begin{aligned} W\Big(\frac{1}{\sqrt{2}}(|x_0\rangle + |x_0 \oplus a\rangle)\Big) &= \frac{1}{\sqrt{2}}\Big(\frac{1}{\sqrt{2^n}}\sum_y((-1)^{x_0\cdot y} + (-1)^{(x_0\oplus a)\cdot y)}\,|y\rangle\Big) \\ &= \frac{1}{\sqrt{2^{n+1}}}\sum_y(-1)^{x_0\cdot y}(1 + (-1)^{a\cdot y})\,|y\rangle \\ &= \frac{2}{\sqrt{2^{n+1}}}\sum_{y\cdot a\ \text{even}}(-1)^{x_0\cdot y}\,|y\rangle \end{aligned}$$



Measurement yields random y such that $y \cdot a = 0 \mod 2$, so the unknown bits of a_i of a must satisfy this equation:

$$y_0 \cdot a_0 \oplus \cdots \oplus y_{n-1} \cdot a_{n-1} = 0$$

Computation is repeated until n linearly independent equations have been found. Each time, the resulting equation has at least a 50% probability of being linearly independent of the previous equations. After repeating $2n$ times, there is a 50% chance that n linearly independent equations have been found. These equations can be solved to find a in $O(n^2)$ steps.

Summary

- Any efficient reversible classical circuit can be efficiently implemented as a quantum circuit.
  - Use inverse function to reduce space and unentangle temporary bits.
- For quantum advantage, add some non-classical operations.
  - E.g, phase change.
- Are these algorithms really useful?
  - Perhaps not directly, but they illustrate ways in which quantum computing may have an advantage over classical computing.