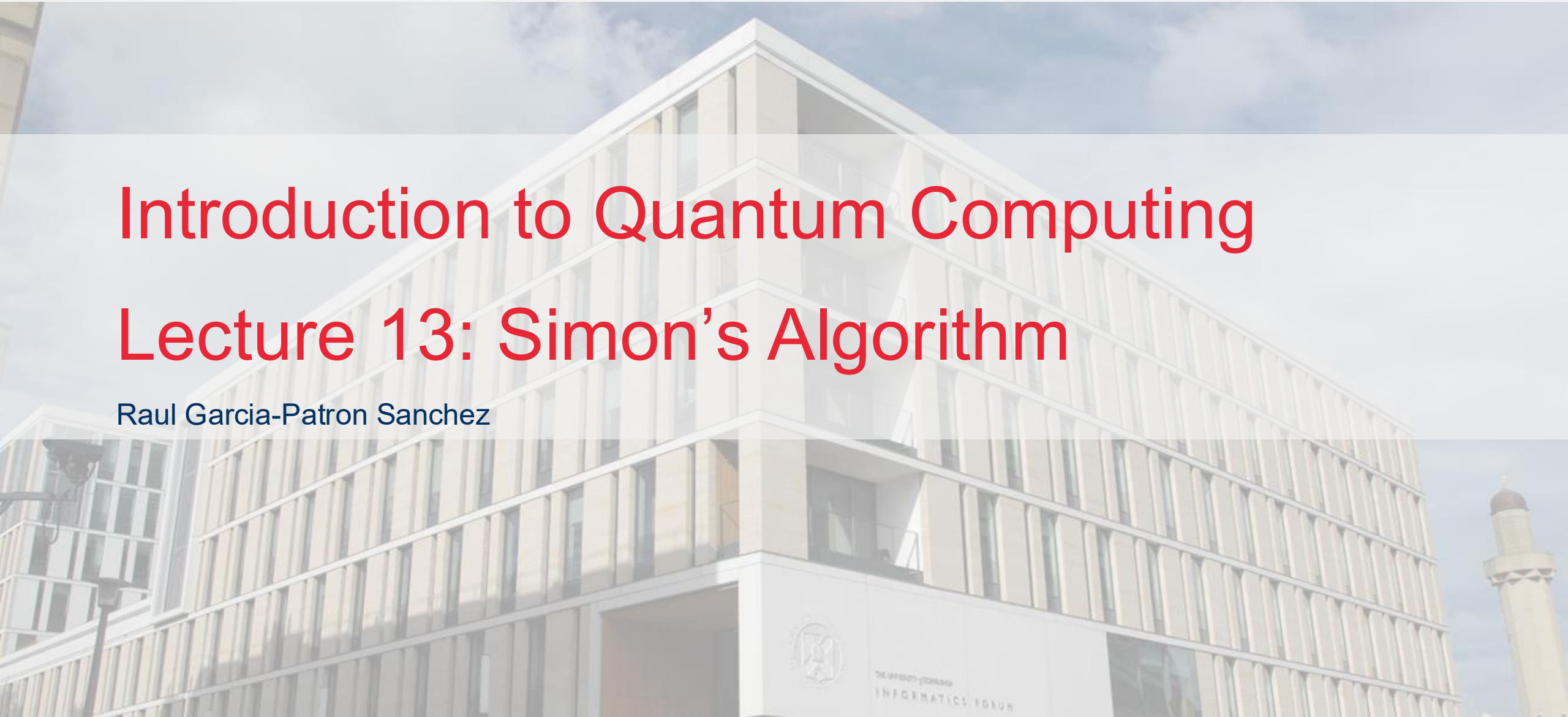


THE UNIVERSITY of EDINBURGH
informatics

Introduction to Quantum Computing

Lecture 13: Simon's Algorithm

Raul Garcia-Patron Sanchez

Find the period

$f : \{0,1\}^n \to \{0,1\}^n$

Promise: f is 2-to-1 and periodic on $\{0,1\}^n$

$f(x) = f(x') \Leftrightarrow x = x' \text{ or } x = x' \oplus a \quad (a \neq 0)$

Problem: Find a

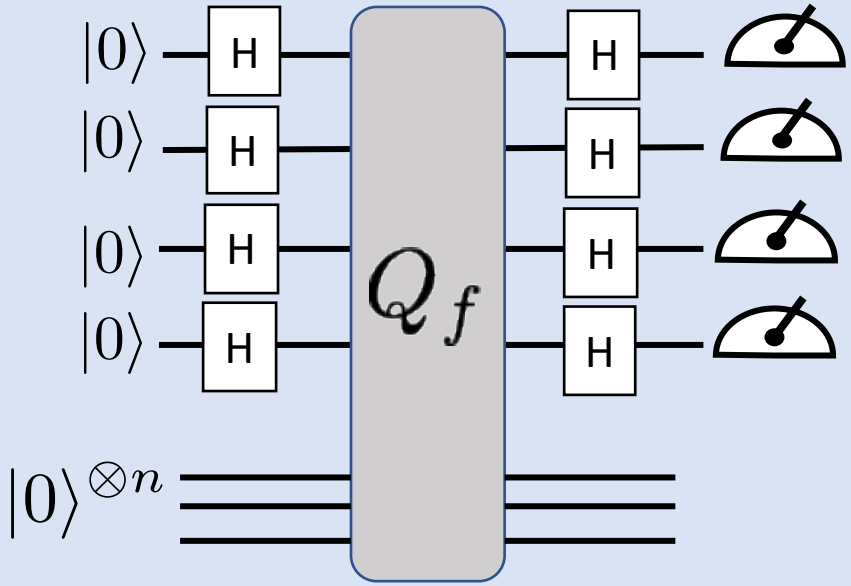
Exponential separation



- Inspired Shor's Algorithm
- Can be used to break some cryptographic primitives (see references)

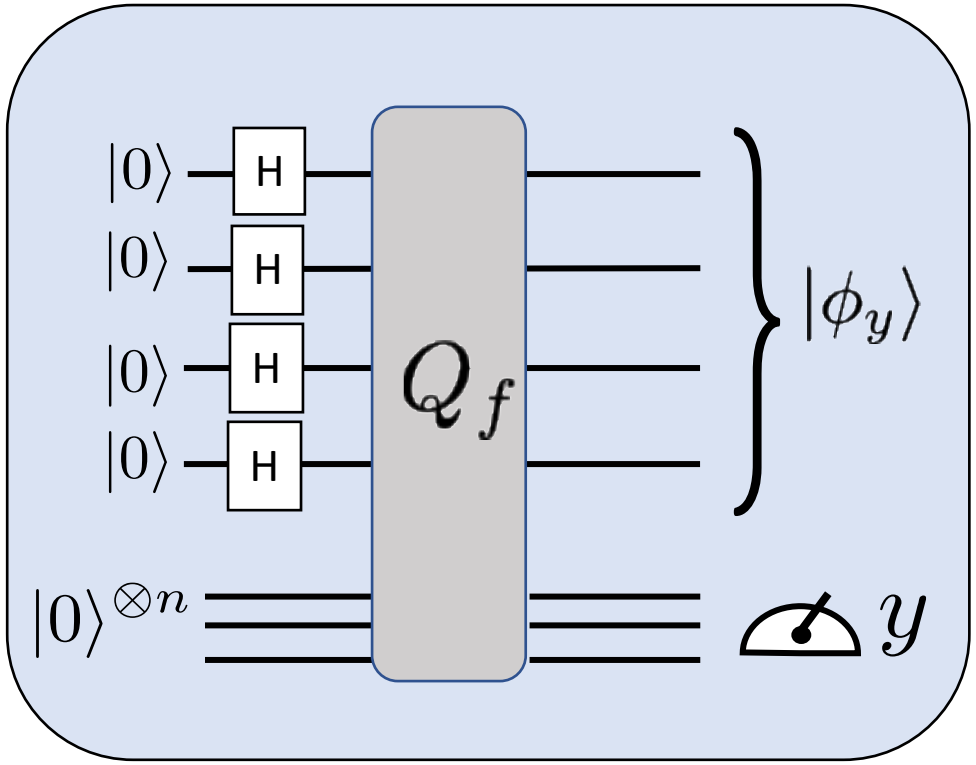
$$|x\rangle|0^n\rangle \xrightarrow{Q_f} |x\rangle|f(x)\rangle$$

Quantum subroutine



Simon's Algorithm

$$f : \{0,1\}^n \to \{0,1\}^n$$
$$f(x) = f(x') \Leftrightarrow x = x' \text{ or } x = x' \oplus a$$



- $|0^n\rangle \otimes |0^n\rangle \xrightarrow{H^{\otimes n} \otimes I_{2^n}} \left(\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle \right) \otimes |0^n\rangle$

$$\xrightarrow{Q_f} \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle |f(x)\rangle$$

$$= \frac{1}{\sqrt{2^n}} \sum_{\text{Image}} \left(\sum_{\text{Preimage of } f(x)} |x\rangle \right) |f(x)\rangle$$

- Measurement outcome y : $|\phi_y\rangle = \frac{1}{\sqrt{2}} \left(|x_y\rangle + |x_y \oplus a\rangle \right)$



Projective and partial measurements

Projective measurement

A projective measurement consists of a set of projectors Π_i

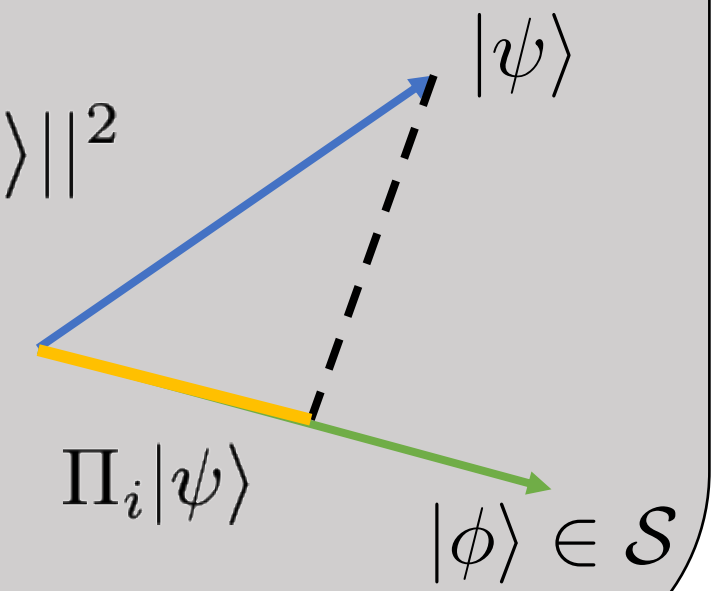
Completeness relation: $\sum_{i=0}^{l} \Pi_i = I_d$

Orthogonality relation: $\Pi_m \Pi_n = \delta_{n,m} \Pi_m$

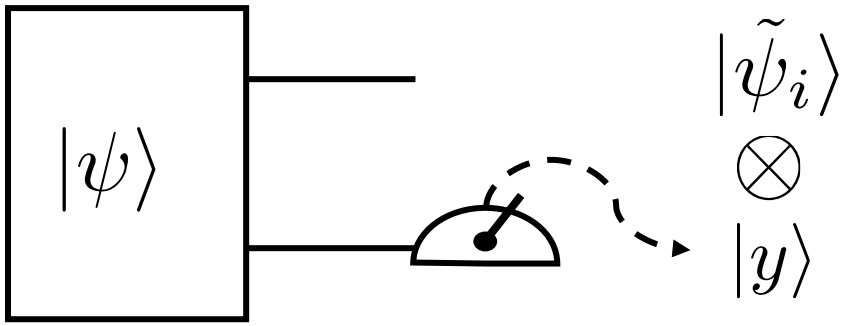
Probability of outcome i reads: $P(i) = ||\Pi_i|\psi\rangle||^2$

The quantum state is updated to

$$|\psi_i\rangle = \frac{\Pi_i|\psi\rangle}{||\Pi_i|\psi\rangle||}$$



Two-qubit example



$$\tilde{\Pi}_0 = I \otimes |0\rangle\langle 0|$$
$$\tilde{\Pi}_1 = I \otimes |1\rangle\langle 1|$$
$$\tilde{\Pi}_0 + \tilde{\Pi}_1 = I$$

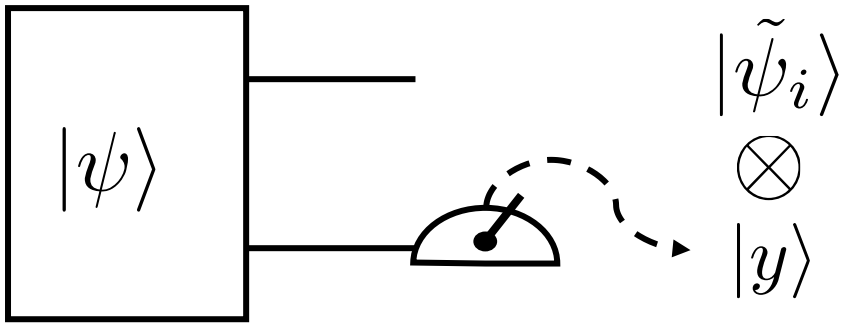
$$|\psi\rangle = \psi_{00}|0\rangle \otimes |0\rangle + \psi_{01}|0\rangle \otimes |1\rangle + \psi_{10}|1\rangle \otimes |0\rangle + \psi_{11}|1\rangle \otimes |1\rangle$$

$$\tilde{\Pi}_0|i\rangle \otimes |0\rangle = (I \otimes |0\rangle\langle 0|)(|i\rangle \otimes |0\rangle) = I|i\rangle \otimes |0\rangle \underbrace{\langle 0|0\rangle}_{=1} = |i\rangle \otimes |0\rangle$$

$$\tilde{\Pi}_0|i\rangle \otimes |1\rangle = (I \otimes |0\rangle\langle 0|)(|i\rangle \otimes |1\rangle) = I|i\rangle \otimes |0\rangle \underbrace{\langle 0|1\rangle}_{=0} = 0$$

$$\tilde{\Pi}_0|\psi\rangle = \psi_{00}|0\rangle \otimes |0\rangle + \psi_{10}|1\rangle \otimes |0\rangle = (\psi_{00}|0\rangle + \psi_{10}|1\rangle) \otimes |0\rangle$$

Two-qubit example



$$\tilde{\Pi}_0 = I \otimes |0\rangle\langle 0|$$
$$\tilde{\Pi}_1 = I \otimes |1\rangle\langle 1|$$
$$\tilde{\Pi}_0 + \tilde{\Pi}_1 = I$$

$$|\psi\rangle = \psi_{00}|0\rangle \otimes |0\rangle + \psi_{01}|0\rangle \otimes |1\rangle + \psi_{10}|1\rangle \otimes |0\rangle + \psi_{11}|1\rangle \otimes |1\rangle$$

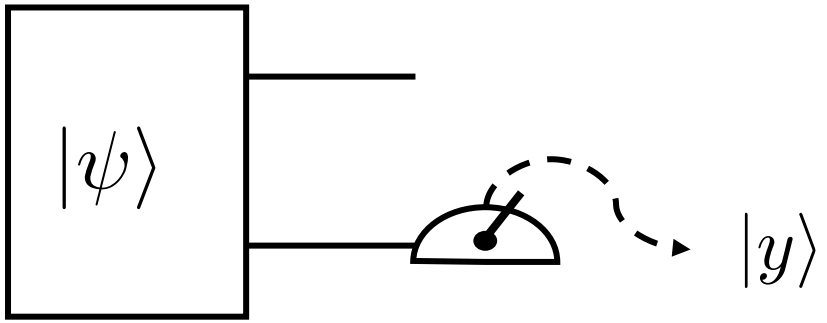
$$\tilde{\Pi}_0|\psi\rangle = \psi_{00}|0\rangle \otimes |0\rangle + \psi_{10}|1\rangle \otimes |0\rangle = (\psi_{00}|0\rangle + \psi_{10}|1\rangle) \otimes |0\rangle$$

$$||\tilde{\Pi}_0|\psi\rangle||^2 = |\psi_{00}|^2 + |\psi_{10}|^2$$

$$|\psi_0\rangle = \frac{\tilde{\Pi}_0|\psi\rangle}{||\tilde{\Pi}_0|\psi\rangle||} = \frac{1}{\sqrt{|\psi_{00}|^2 + |\psi_{10}|^2}} (\psi_{00}|0\rangle + \psi_{10}|1\rangle) \otimes |0\rangle = |\tilde{\psi}_0\rangle \otimes |0\rangle$$

Two-register example

Two registers of n and m qubits respectively :



$$\tilde{\Pi}_y = I_{2^n} \otimes |y\rangle\langle y|$$

$$\sum_{y\in\{0,1\}^m} \Pi_y = I_{2^m} \text{ and } \sum_{y\in\{0,1\}^m} \tilde{\Pi}_y = I_{2^{n+m}}$$

$$|\psi\rangle = \sum_{x,w} \psi_{x,w}|x\rangle \otimes |w\rangle$$

Projected vector

$$\tilde{\Pi}_y|\psi\rangle = (I_{2^n} \otimes |y\rangle\langle y|)|\psi\rangle = (\sum_x \psi_{x,y}|x\rangle) \otimes |y\rangle$$

Probability of outcome “y”: $||\tilde{\Pi}_y|\psi\rangle||^2 = \sum_x |\psi_{x,y}|^2$

Updated state after outcome “y”:

$$|\psi_y\rangle = \frac{\tilde{\Pi}_y|\psi\rangle}{||\tilde{\Pi}_y|\psi\rangle||} = \frac{1}{\sqrt{\sum_x |\psi_{x,y}|^2}} \left(\sum_x \psi_{x,y}|x\rangle\right) \otimes |y\rangle = |\tilde{\psi}_y\rangle \otimes |y\rangle$$

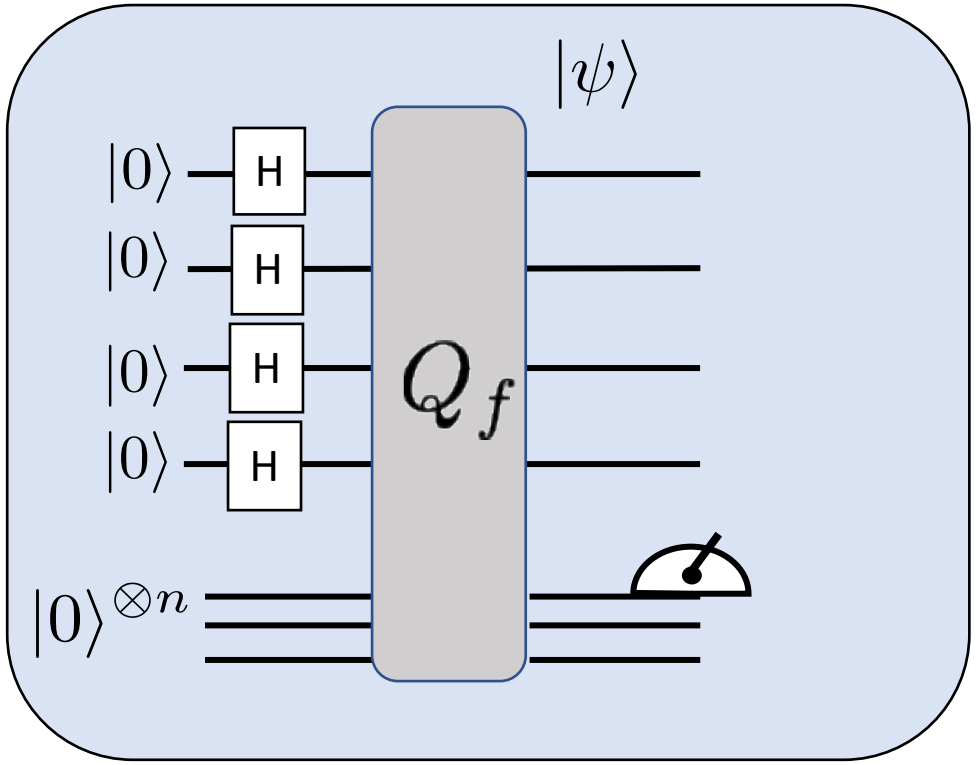


Simon’s Algorithm

Raul Garcia-Patron Sanchez

Simon's Algorithm

$$f : \{0,1\}^n \to \{0,1\}^n$$
$$f(x) = f(x') \Leftrightarrow x = x' \text{ or } x = x' \oplus a$$



- $|0^n\rangle \otimes |0^n\rangle \xrightarrow{H^{\otimes n} \otimes I_{2^n}} \left(\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle \right) \otimes |0^n\rangle$

$$= \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle \otimes |0^n\rangle$$

$$\xrightarrow{Q_f} \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle |f(x)\rangle$$

$$|\psi\rangle = \frac{1}{\sqrt{2^n}} \sum_{w \in \text{Image}} \left(\sum_{x : f(x) = w} |x\rangle \right) \otimes |w\rangle$$

$$\text{Image} = \{f(x) : x \in \{0,1\}^n\}$$

Simon's Algorithm

$$f : \{0,1\}^n \to \{0,1\}^n$$
$$f(x) = f(x') \Leftrightarrow x = x' \text{ or } x = x' \oplus a$$

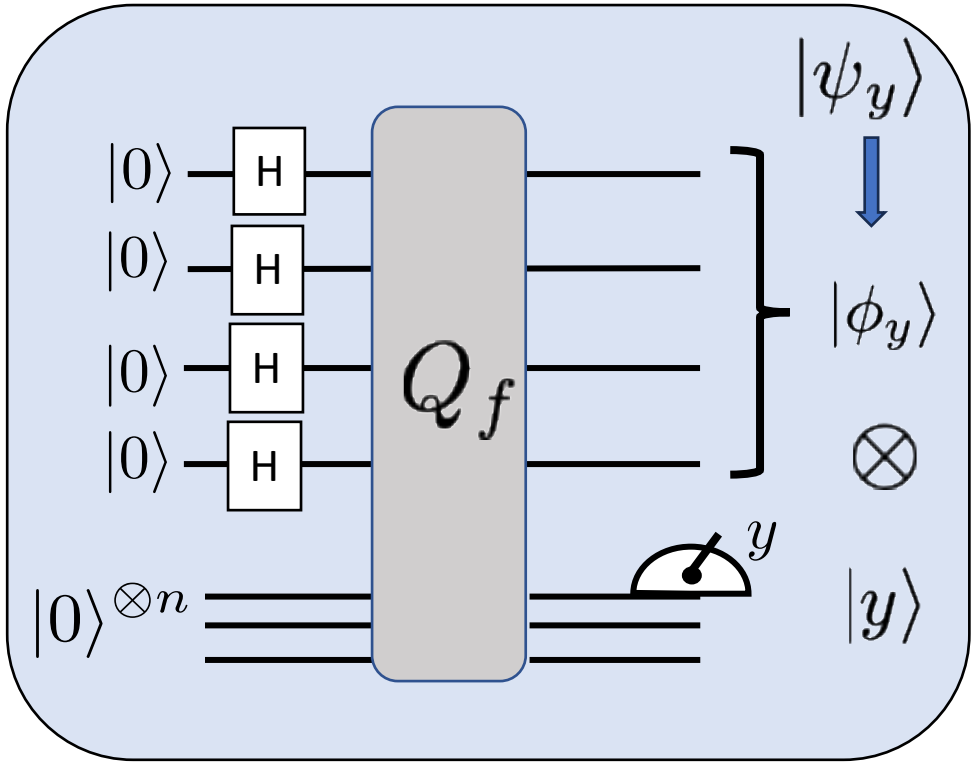
- $$|\psi\rangle = \frac{1}{\sqrt{2^n}} \sum_{w \in \text{Image}} \left(\sum_{x:f(x)=w} |x\rangle \right) \otimes |w\rangle$$

$$\tilde{\Pi}_y |\psi\rangle = \frac{1}{\sqrt{2^n}} \left(\sum_{x:f(x)=y} |x\rangle \right) \otimes |y\rangle$$

$$||\tilde{\Pi}_y|\psi\rangle||^2 = \frac{|\{x : f(x) = y\}|}{2^n}$$

$|\mathcal{S}|$: size of set $\mathcal{S}$

$$|\psi_y\rangle = \frac{\tilde{\Pi}_y|\psi\rangle}{||\tilde{\Pi}_y|\psi\rangle||} = \left(\frac{1}{\sqrt{|\{x : f(x) = y\}|}} \sum_{x:f(x)=y} |x\rangle \right) \otimes |y\rangle = |\phi_y\rangle \otimes |y\rangle$$



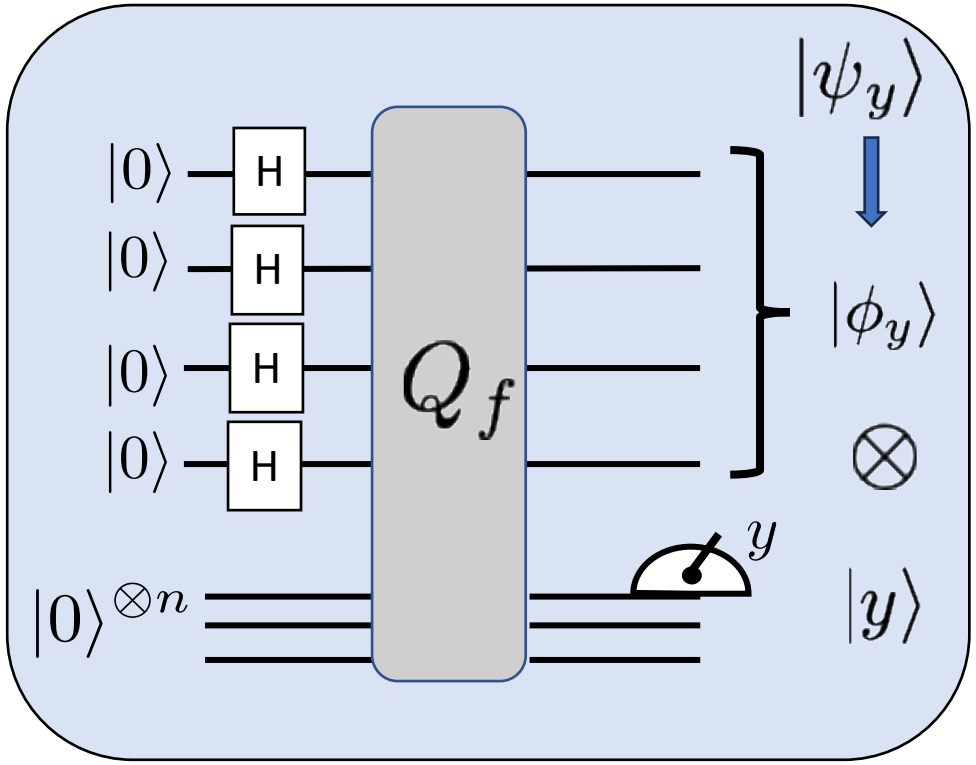
$$\tilde{\Pi}_y = I_{2^n} \otimes |y\rangle\langle y|$$

We are measuring the lower register. Remember we cannot choose the outcome, but it results from a random process. For a given output "y" we can define its corresponding projector (see above). The squared norm of the projected vector gives the probability. We can then compute the updated global state (see left box).

Simon's Algorithm

$$f : \{0,1\}^n \to \{0,1\}^n$$
$$f(x) = f(x') \Leftrightarrow x = x' \text{ or } x = x' \oplus a$$

$$|\psi_y\rangle = \frac{\tilde{\Pi}_y|\psi\rangle}{||\tilde{\Pi}_y|\psi\rangle||} = \left(\frac{1}{\sqrt{|\{x : f(x) = y\}|}} \sum_{x:f(x)=y} |x\rangle\right) \otimes |y\rangle = |\phi_y\rangle \otimes |y\rangle$$



- $f(x)$ being a 2-to-1 function:

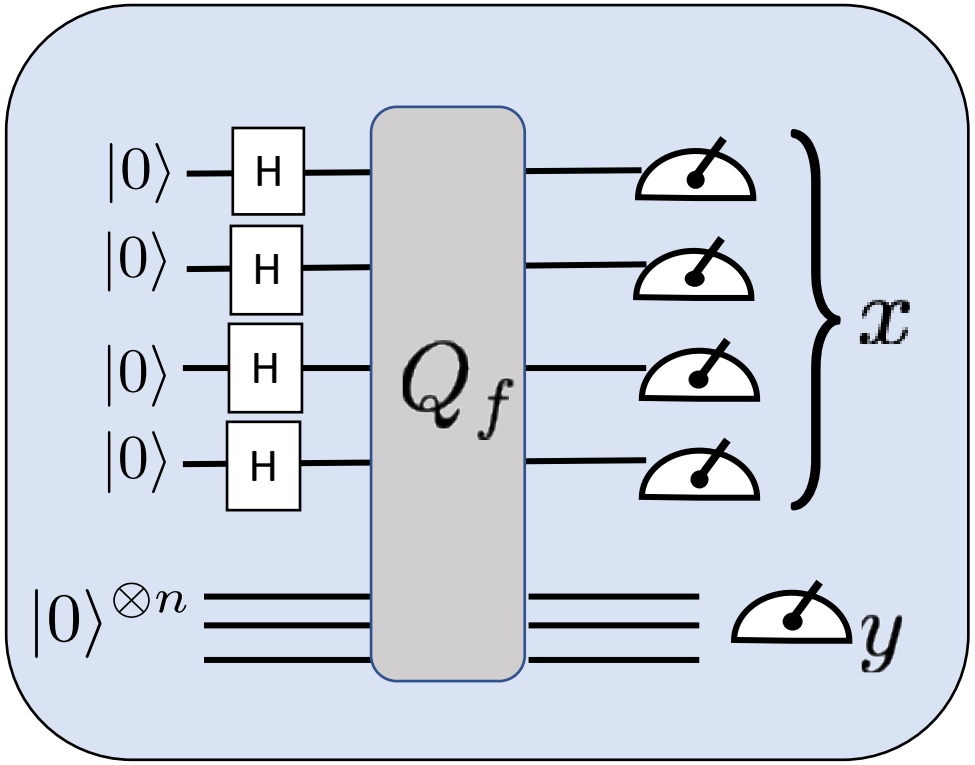
$$|\phi_y\rangle = \frac{1}{\sqrt{2}}\left[|x_y\rangle + |x_y \oplus a\rangle\right]$$

We can think of |y> as the state of the lower register encoding the outcome of the measurement. Remark: in a practical implementation it would most probably be a classical register.

We are not going to act on the lower register anymore after its measurement. Therefore, to analyze the algorithm we could follow two equivalent approaches: 1) use the global state $|\psi_y\rangle$and any operation A (unitary or projector) acting on the upper register needs to be represented by a tensor product A$\otimes I$; 2) Ignore the second register. For simplicity we are going to use option 2 and work with $|\phi_y\rangle$ directly.

What if we measure after the oracle?

$$f : \{0,1\}^n \to \{0,1\}^n$$
$$f(x) = f(x') \Leftrightarrow x = x' \text{ or } x = x' \oplus a$$



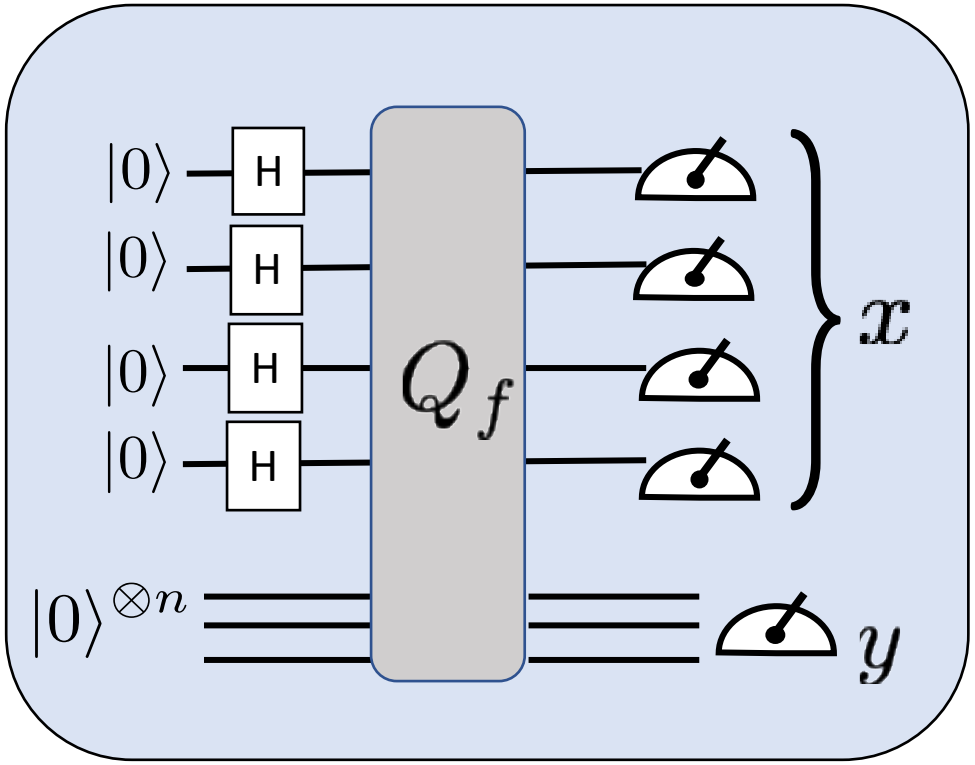
- $f(x)$ being a 2-to-1 function:

$$|\phi_y\rangle = \frac{1}{\sqrt{2}}\left[|x_y\rangle + |x_y \oplus a\rangle\right]$$

?

What if we measure after the oracle?

$$f : \{0,1\}^n \to \{0,1\}^n$$
$$f(x) = f(x') \Leftrightarrow x = x' \text{ or } x = x' \oplus a$$



- $f(x)$ being a 2-to-1 function:

$$|\phi_y\rangle = \frac{1}{\sqrt{2}}\left[|x_y\rangle + |x_y \oplus a\rangle\right]$$

We get an input “x” associated with output “y” as y=f(x). But to guess “a” we need both “x” and “x + a (mod 2)” and the probability to measure the same “y” again is going to be exponentially small.

Simon's Algorithm

$$f(x) = f(x') \Leftrightarrow x = x' \text{ or } x = x' \oplus a$$

- Before last Hadamards: $|\phi_y\rangle = \frac{1}{\sqrt{2}}\left(|x_y\rangle + |x_y \oplus a\rangle\right)$
- Walsh–Hadamard (and linearity):

$$|\phi_y\rangle \xrightarrow{H^{\otimes n}} \frac{1}{\sqrt{2}}\left[\left(\frac{1}{\sqrt{2^n}}\sum_{z\in\{0,1\}^n}(-1)^{x_y\cdot z}|z\rangle\right) + \left(\frac{1}{\sqrt{2^n}}\sum_{z\in\{0,1\}^n}(-1)^{(x_y\oplus a)\cdot z}|z\rangle\right)\right]$$

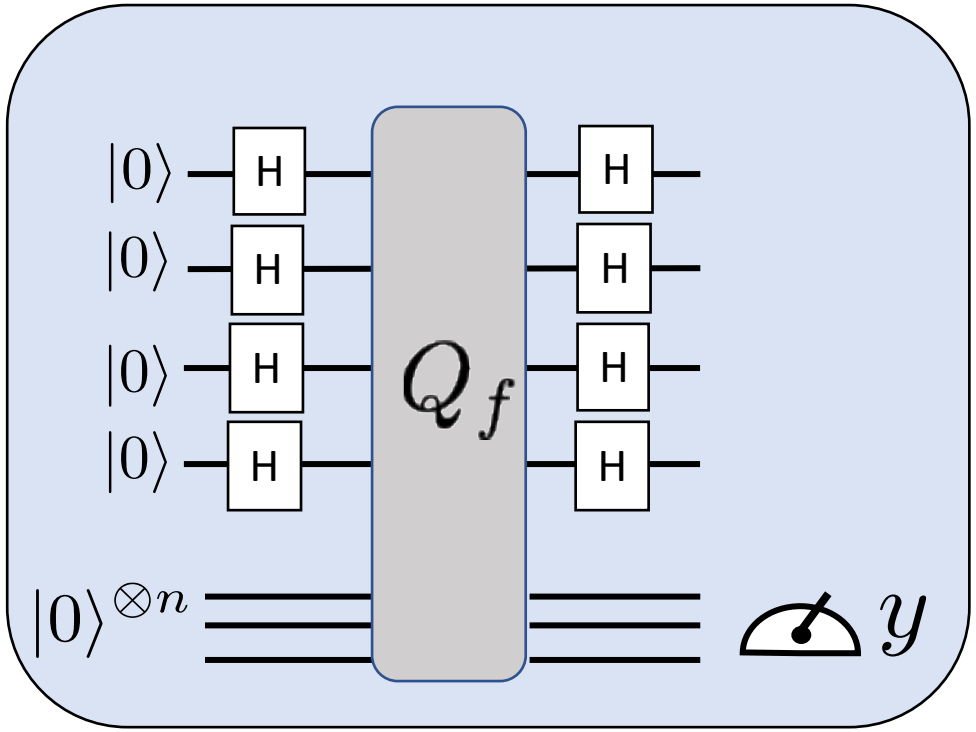
$$= \frac{1}{\sqrt{2^{n+1}}}\sum_{z\in\{0,1\}^n}\left[(-1)^{x_y\cdot z} + (-1)^{(x_y\oplus a)\cdot z}\right]|z\rangle$$

$$= \frac{1}{\sqrt{2^{n+1}}}\sum_{z\in\{0,1\}^n}(-1)^{x_y\cdot z}\left[1 + (-1)^{a\cdot z}\right]|z\rangle$$

$$= \frac{1}{\sqrt{2^{n-1}}}\sum_{z:a\cdot z=0}(-1)^{x_y\cdot z}|z\rangle$$

For z s.t. $a \cdot z = 1 \pmod 2$ we get a zero contribution (amplitude interference). Only z s.t. $a \cdot z = 0 \pmod 2$ are left (with a contribution of 2), which changes the $1/\sqrt{2^{n+1}}$ to $1/\sqrt{2^{n-1}}$ in the normalization.

There exist 2^{n-1} n-bit strings z satisfying $a \cdot z = 0 \pmod 2$, which is consistent with the normalization term as one would expect.



Walsh–Hadamard transform

$$|x\rangle \xrightarrow{H^{\otimes n}} \frac{1}{\sqrt{2^n}}\sum_{z\in\{0,1\}^n}(-1)^{x\cdot z}|z\rangle$$

Simon's Algorithm

$$f(x) = f(x') \Leftrightarrow x = x' \text{ or } x = x' \oplus a$$

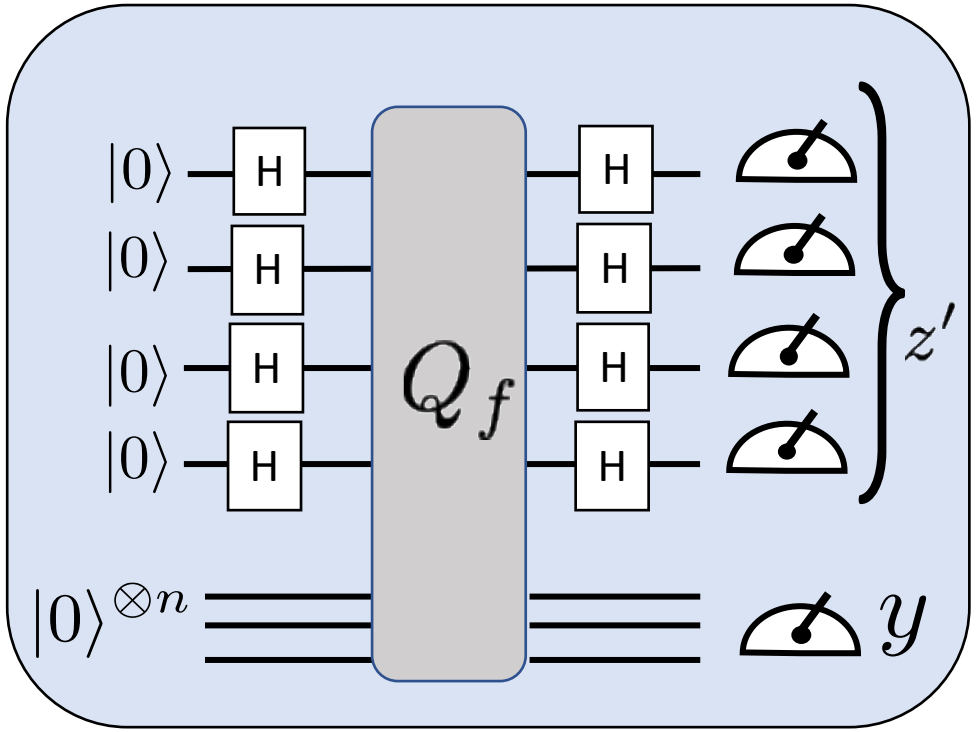
- Before last Hadamards: $|\phi_y\rangle = \frac{1}{\sqrt{2}}\left(|x_y\rangle + |x_y \oplus a\rangle\right)$
- Before measurement: $|\phi'_y\rangle = \frac{1}{\sqrt{2^{n-1}}} \sum_{z:a\cdot z=0} (-1)^{x_y \cdot z} |z\rangle$

Because we are measuring the computational basis on the full upper register (remember we are ignoring the lower register) we can use the old rules for outcome probabilities:

Probability of outcome z' : $P(z') = |\langle z'|\phi'_y\rangle|^2 = \frac{1}{2^{n-1}}$

$z' \in \{0,1\}^n : a \cdot z' = 0$

$\{z' \in \{0,1\}^n : a \cdot z' = 1\}$ have zero amplitude and probability.



Notation remark: We have used so far z to label the elements of the computational basis of the upper register state before measurement. We are using z' to refer to the label of the measurement outcome (in the computational basis).

$$\{z' \in \{0,1\}^n : a \cdot z' = 0\}$$

Each oracle call generates a sample z' of n bits that satisfies: $a \cdot z' = 0 \pmod 2$.

Simon's Algorithm

$$f(x) = f(x') \Leftrightarrow x = x' \text{ or } x = x' \oplus a$$

- Classical post-processing: solve system of linear equations:

$$\begin{cases} a_1 z_1^{(1)} + a_2 z_2^{(1)} + \cdots + a_n z_n^{(1)} = 0 \bmod 2 \\ a_1 z_1^{(2)} + a_2 z_2^{(2)} + \cdots + a_n z_n^{(2)} = 0 \bmod 2 \\ \vdots \\ a_1 z_1^{(n-1)} + a_2 z_2^{(n-1)} + \cdots + a_n z_n^{(n-1)} = 0 \bmod 2 \end{cases}$$

(For simplicity we use z instead of z' from now)

1. The i-th outcome generates a sample $z^{(i)}$ of n bits that satisfies: $a \cdot z^{(i)} = 0 \pmod 2$
2. We generate $n-1$ samples.
3. If $n-1$ equations are independent we solve the system of linear equations using row-reduction with XORs to triangular form.

- $n-1$ independent equations leave us with a 1-dimensional space of solutions $\{0, a\}$

  We were given the promise that $a \neq 0$, so only a is left.

- Probability of $n-1$outcomes being all linearly independent (see proof slide 20): $P > \frac{1}{4}$

Repeat until success (getting n-1 lin. ind. eq.) adds a constant factor to the required oracle calls guaranteeing the O(n) scaling.
A better strategy is to add sample $z^{(i)}$ only if it is linearly independent of the previous ones and keep trying until you collect n-1 .
For some hardware it may be better to do all quantum runs before classical post-processing. Then generate $\sim 4n$ samples $z^{(i)}$ and extract $n-1$ lin. Ind. ones.

$O(n)$ oracle calls

$O(n^3)$ classical post-processing

Classical queries

$$f : \{0,1\}^n \to \{0,1\}^n$$
$$f(x) = f(x') \Leftrightarrow x = x' \text{ or } x = x' \oplus a$$

- We need to find a pair x and x' such that $f(x) = f(x')$

  Solution $a = x \oplus x'$

- Success probability: $\Pr[x = x' \oplus a] = \dfrac{1}{2^n - 1}$

- For T queries we have at most T^2 different pairs:
  - $\Pr[\text{ find } a] \leq T^2 2^{-n}$
  - $T \geq \sqrt{P} 2^{n/2}$

Quantum: $O(n)$

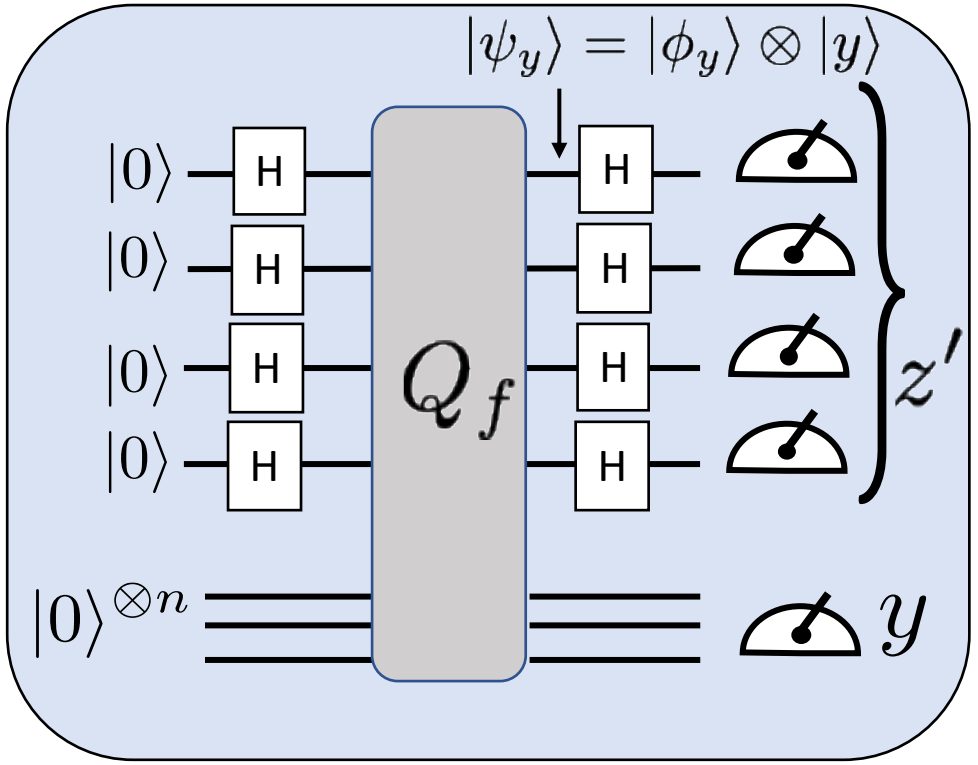
Classical: $\Omega(2^{n/2})$



We do not need to measure the lower register

- Before measurement: $|\phi'_y\rangle = \frac{1}{\sqrt{2^{n-1}}} \sum_{z:a\cdot z=0} (-1)^{x_y \cdot z} |z\rangle$
- Measurement outcome z' satisfies:

$$\{z' \in \{0,1\}^n : a \cdot z' = 0\}$$



Neither "x_y" nor "y" plays a role in the post-processing.
We could forget the value of "y" and nothing would have changed!

We could have done an alternative proof where we were not measuring the lower register, but only the upper one at the end. We would have ended with a "garbage" quantum state on the lower register (with some dependency on the outcome z') that is not needed at any stage of the algorithm.

Remark: in quantum mechanics, measuring a register and forgetting its outcome is equivalent to "removing" the register you would have measured. This can be proven using the density-matrix formalism and partial trace. This formalism is not covered in IQC but you will use it in QCS. See also NC 2.4.

Simon's Algorithm

- $\{z \in \{0,1\}^n : a \cdot z = 0\}$ is of size $2^n/2 = 2^{n-1}$
- the span of k lin. indep. $z^{(i)}$ has size 2^k

- Probability of a linearly independent set from first $n-1$ samples:

$$P = \prod_{k=0}^{n-2} \left(\frac{2^{n-1} - 2^k}{2^{n-1}} \right) = \prod_{j=1}^{n-1} \left(1 - \frac{1}{2^j} \right)$$

Relabeling: $j = n - k - 1$

$$= \prod_{j=1}^{n-1} \left(1 - \frac{1}{2^j} \right) \geq \prod_{j=1}^{\infty} \left(1 - \frac{1}{2^j} \right)$$

Extend the product with terms < 1

Use: $(1-b)(1-c) \geq 1-(b+c)$

$$= \frac{1}{2} \prod_{j=2}^{\infty} \left(1 - \frac{1}{2^j} \right) > \frac{1}{2} \left[1 - \sum_{j=2}^{\infty} \frac{1}{2^j} \right]$$

Geometric series

$$> \frac{1}{2} \left[1 - \frac{1}{4} \sum_{j=0}^{\infty} \frac{1}{2^j} \right] = \frac{1}{4}$$

References

Reading references

1. Simon: RdW 3 and G 8

NC ≣ Michael Nielsen and Isaac Chuang, Quantum Computation and Quantum Information Cambridge University Press (2010)
RdW ≣ Quantum Computing Lecture Notes, Ronald de Wolf, https://arxiv.org/abs/1907.09415
G ≣ Introduction to Quantum Computation, Sevag Gharibian, Lecture notes

Simon's Algorithm breaks cryptographic primitives

1. T. Santoli and C. Schaffner, *Using Simon's algorithm to attack symmetric-key cryptographic primitives*, Quantum Information & Computation 17, 65 (2017).
2. H. Kuwakado and M. Morii, *Quantum distinguisher between the 3-round Feistel cipher and the random permutation,* In 2010 IEEE International Symposium on Information Theory, pages 2682–2685, June (2010).
3. M. Kaplan, G. Leurent, A. Leverrier, and M. Naya-Plascencia, *Breaking symmetric cryptosystems using quantum period finding,* In Advances in Cryptology - CRYPTO 2016. volume 9815 of Lecture Notes in Computer Science, 2016.