Two Basic Quantum Paradigms: Eigenvalue Estimation & Amplitude Amplification*

*** But not necessarily in that order**

Richard Cleve

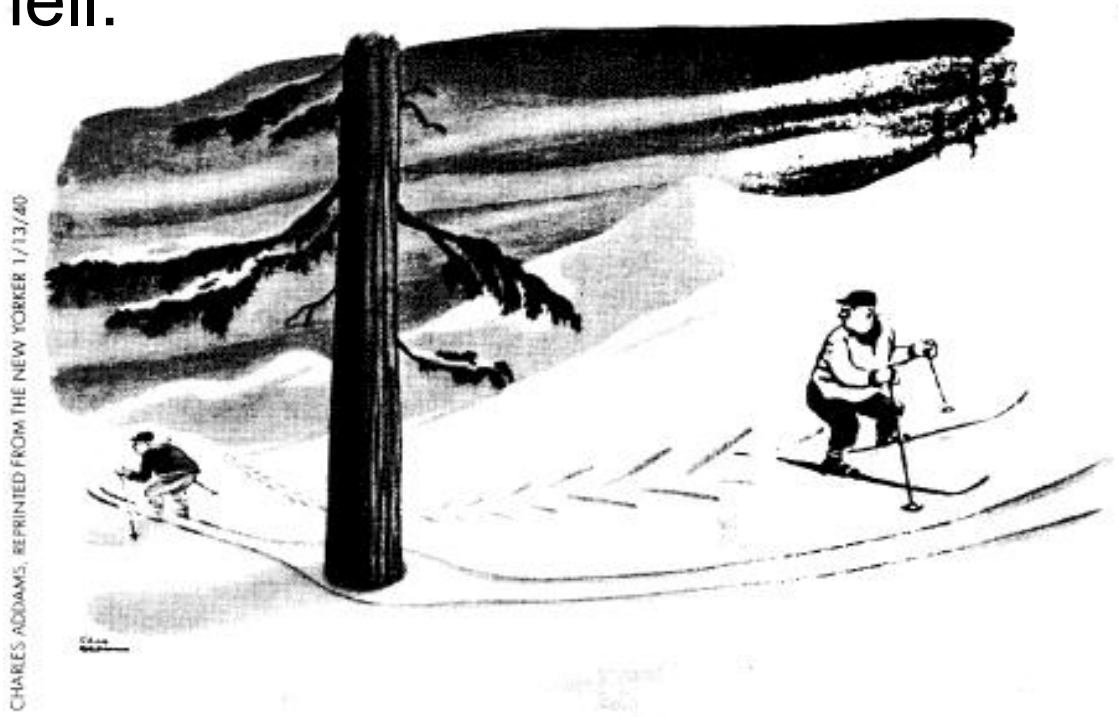
Institute for Quantum Computing & School of Computer Science

University of Waterloo

Fields Institute, Toronto, October 27, 2014

How do quantum algorithms work?

In a nutshell:



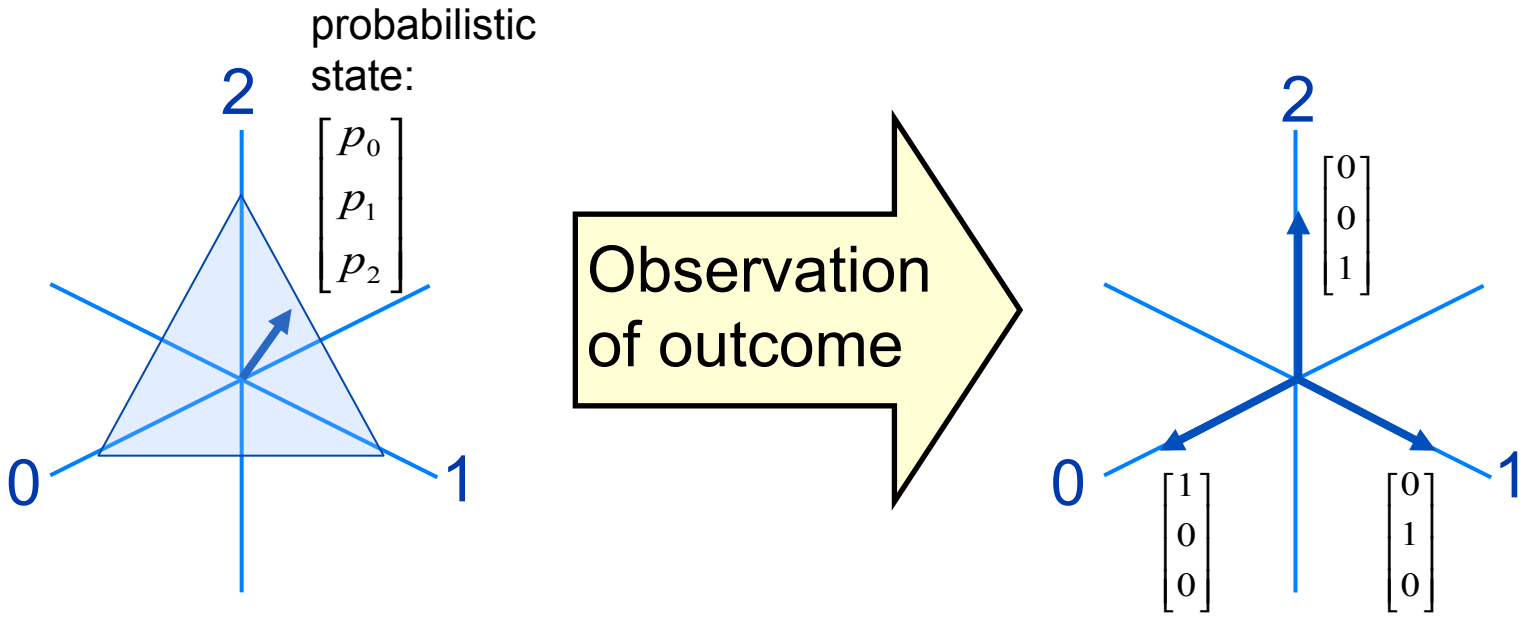
But this isn't "quantum parallelism" in the sense of quantum computers simply performing several computations at once (that approach doesn't work)

Probability

Probability vector:

$\forall x \in \Omega, \quad p_x \geq 0 \quad$ and $\quad \sum_x p_x = 1$

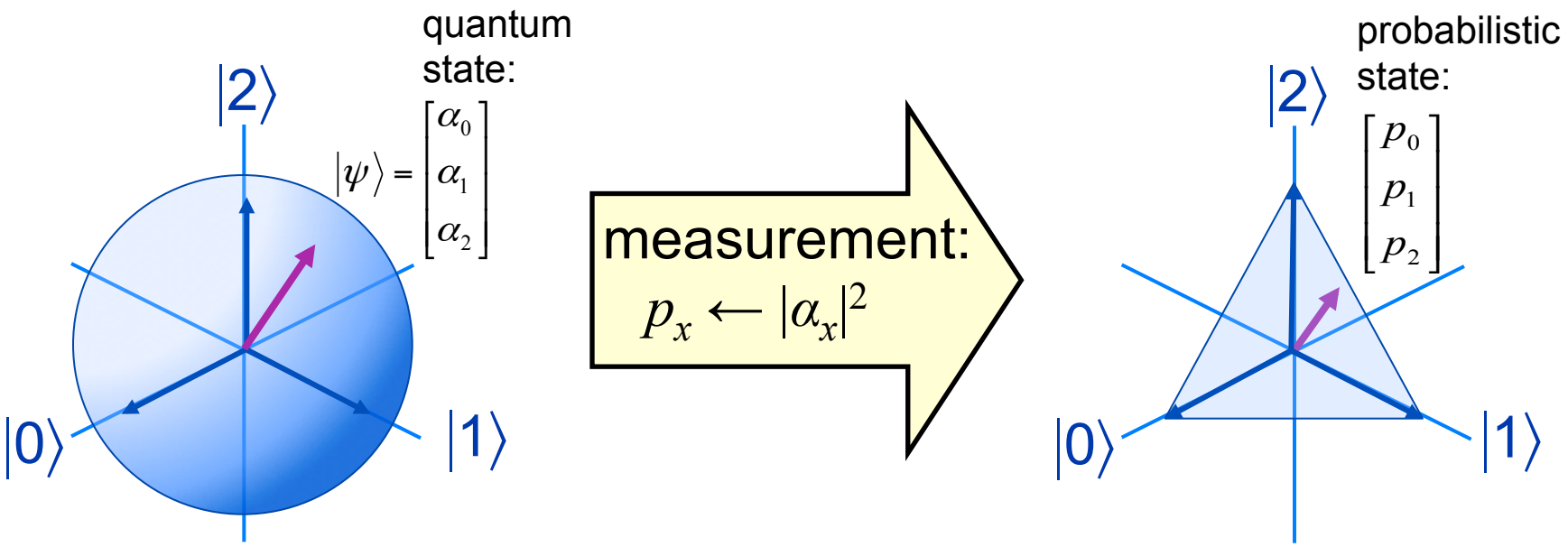
Example: for $\Omega = \{0,1,2\}$



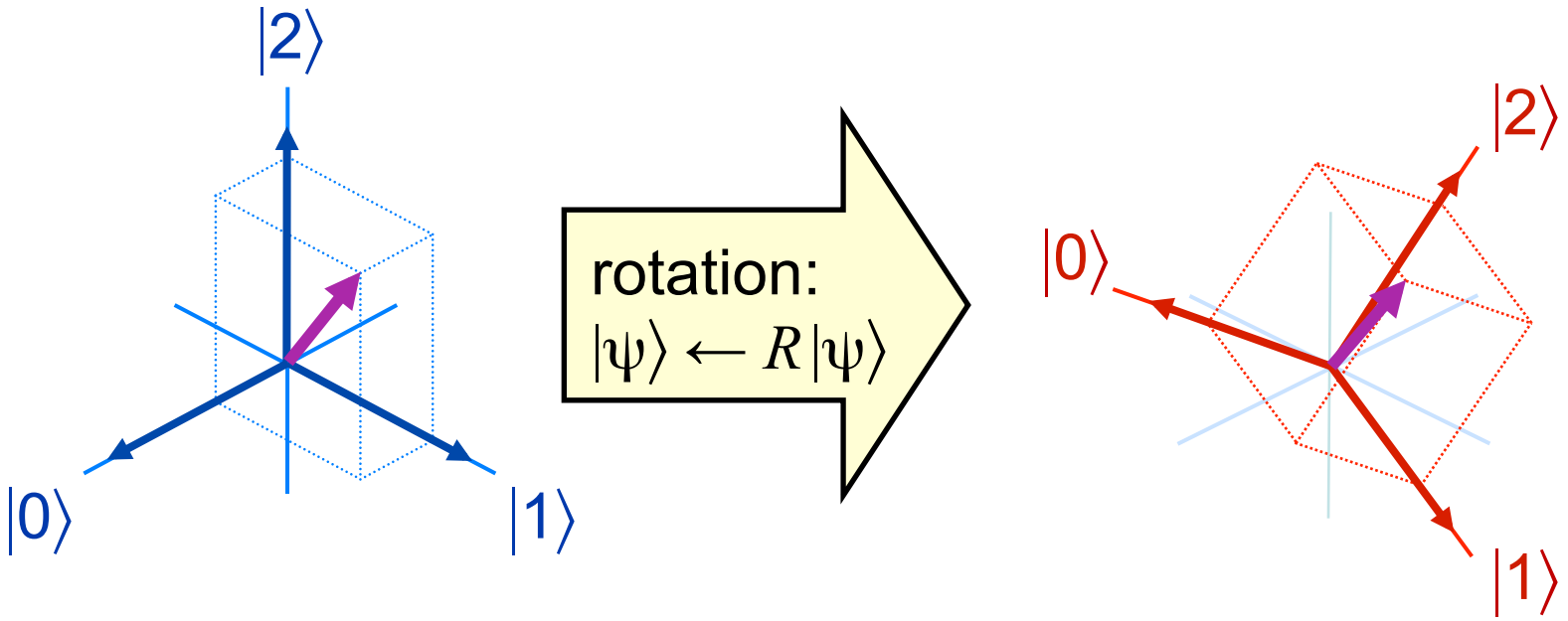
Quantum information framework

Probability *amplitudes*:

$$\forall x \in \Omega,\ \alpha_x \in \mathbb{C} \quad \text{and} \quad \sum_x |\alpha_x|^2 = 1$$



Can rotate the coordinate system!



Different rotations may result in different probability distributions

n-qubit systems

Probabilistic states:

$$\forall x,\ p_x \geq 0$$

$$\sum_x p_x = 1$$

$$\begin{bmatrix} p_{000} \\ p_{001} \\ p_{010} \\ p_{011} \\ p_{100} \\ p_{101} \\ p_{110} \\ p_{111} \end{bmatrix}$$

Quantum states:

$$\forall x,\ \alpha_x \in \mathbb{C}$$

$$\sum_x |\alpha_x|^2 = 1$$

$$\begin{bmatrix} \alpha_{000} \\ \alpha_{001} \\ \alpha_{010} \\ \alpha_{011} \\ \alpha_{100} \\ \alpha_{101} \\ \alpha_{110} \\ \alpha_{111} \end{bmatrix}$$

Dirac notation: $|000\rangle$, $|001\rangle$, $|010\rangle$, …, $|111\rangle$ are basis vectors

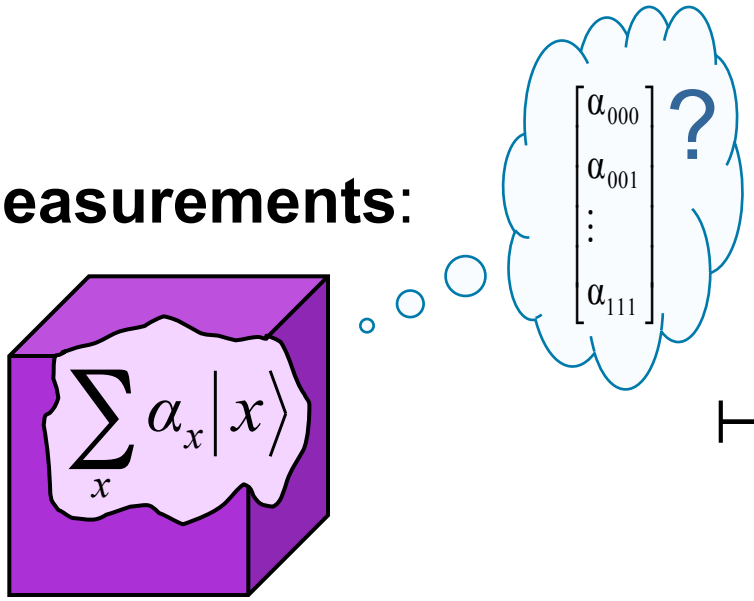
$$|\psi\rangle = \sum_x \alpha_x |x\rangle$$

Operations on *n*-qubit states

Unitary operations: $\sum_x \alpha_x |x\rangle \quad \mapsto \quad U\left(\sum_x \alpha_x |x\rangle\right)$

$(U^\dagger U = I)$

Measurements:



$\mapsto$

$$\begin{cases} 000 & \text{with prob } |\alpha_{000}|^2 \\ 001 & \text{with prob } |\alpha_{001}|^2 \\ \vdots & \vdots \\ 111 & \text{with prob } |\alpha_{111}|^2 \end{cases}$$

… and the quantum state collapses

Basic operations on qubits (I)

(0) Initialize qubit to $|0\rangle$ or to $|1\rangle$

(1) Apply a unitary operation U (unitary means $U^\dagger U = I$)

conjugate transpose

Examples:

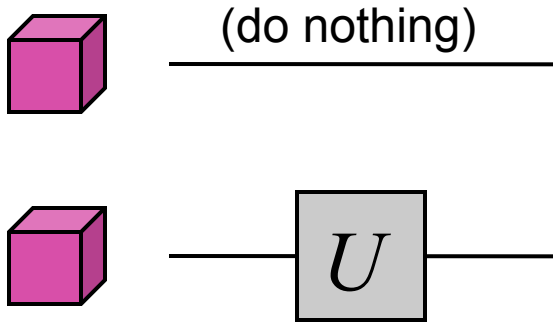
Rotation: $\begin{bmatrix} cos\,\theta & -sin\,\theta \\ sin\,\theta & cos\,\theta \end{bmatrix}$

NOT (bit flip): $\sigma_x = X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$

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

Phase flip: $\sigma_z = Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$

Example of a one-qubit gate applied to a two-qubit system



$$U = \begin{bmatrix} u_{00} & u_{01} \\ u_{10} & u_{11} \end{bmatrix}$$

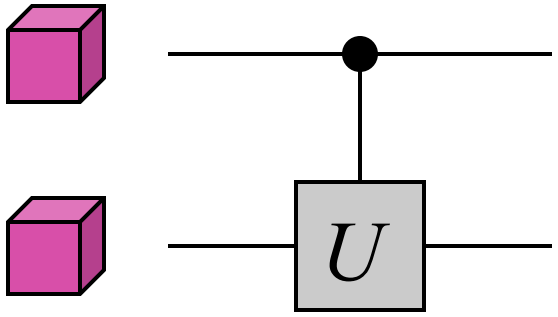
Maps basis states as:

$$|0\rangle|0\rangle \rightarrow |0\rangle U|0\rangle$$
$$|0\rangle|1\rangle \rightarrow |0\rangle U|1\rangle$$
$$|1\rangle|0\rangle \rightarrow |1\rangle U|0\rangle$$
$$|1\rangle|1\rangle \rightarrow |1\rangle U|1\rangle$$

The resulting 4x4 matrix is

$$I \otimes U = \begin{bmatrix} u_{00} & u_{01} & 0 & 0 \\ u_{10} & u_{11} & 0 & 0 \\ 0 & 0 & u_{00} & u_{01} \\ 0 & 0 & u_{10} & u_{11} \end{bmatrix}$$

Controlled-U gates



$$U = \begin{bmatrix} u_{00} & u_{01} \\ u_{10} & u_{11} \end{bmatrix}$$

Resulting 4x4 matrix is controlled-U =

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & u_{00} & u_{01} \\ 0 & 0 & u_{10} & u_{11} \end{bmatrix}$$

Maps basis states as:

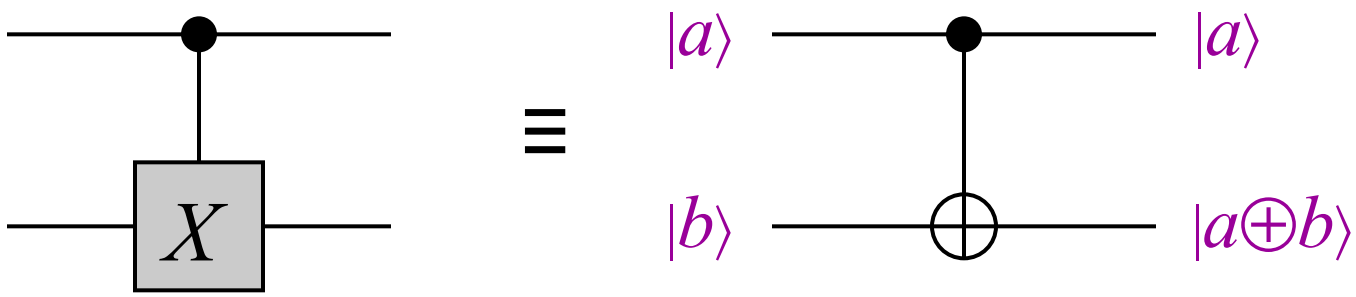
$|0\rangle|0\rangle \rightarrow |0\rangle|0\rangle$

$|0\rangle|1\rangle \rightarrow |0\rangle|1\rangle$

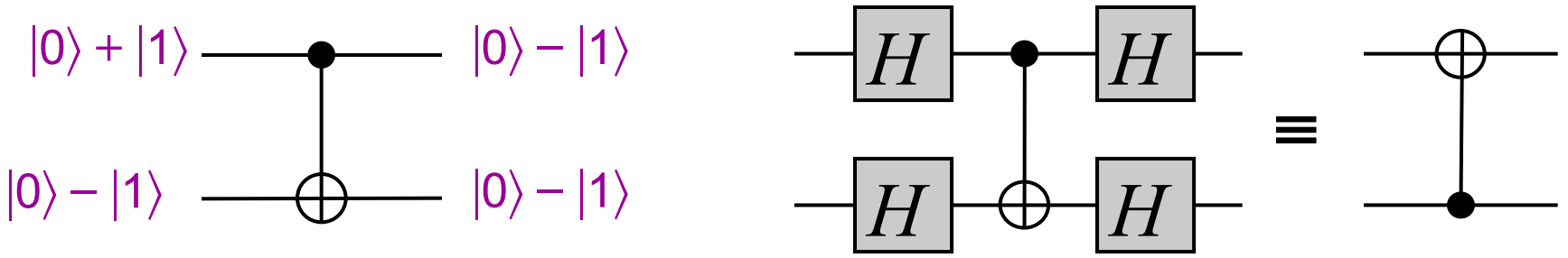
$|1\rangle|0\rangle \rightarrow |1\rangle U|0\rangle$

$|1\rangle|1\rangle \rightarrow |1\rangle U|1\rangle$

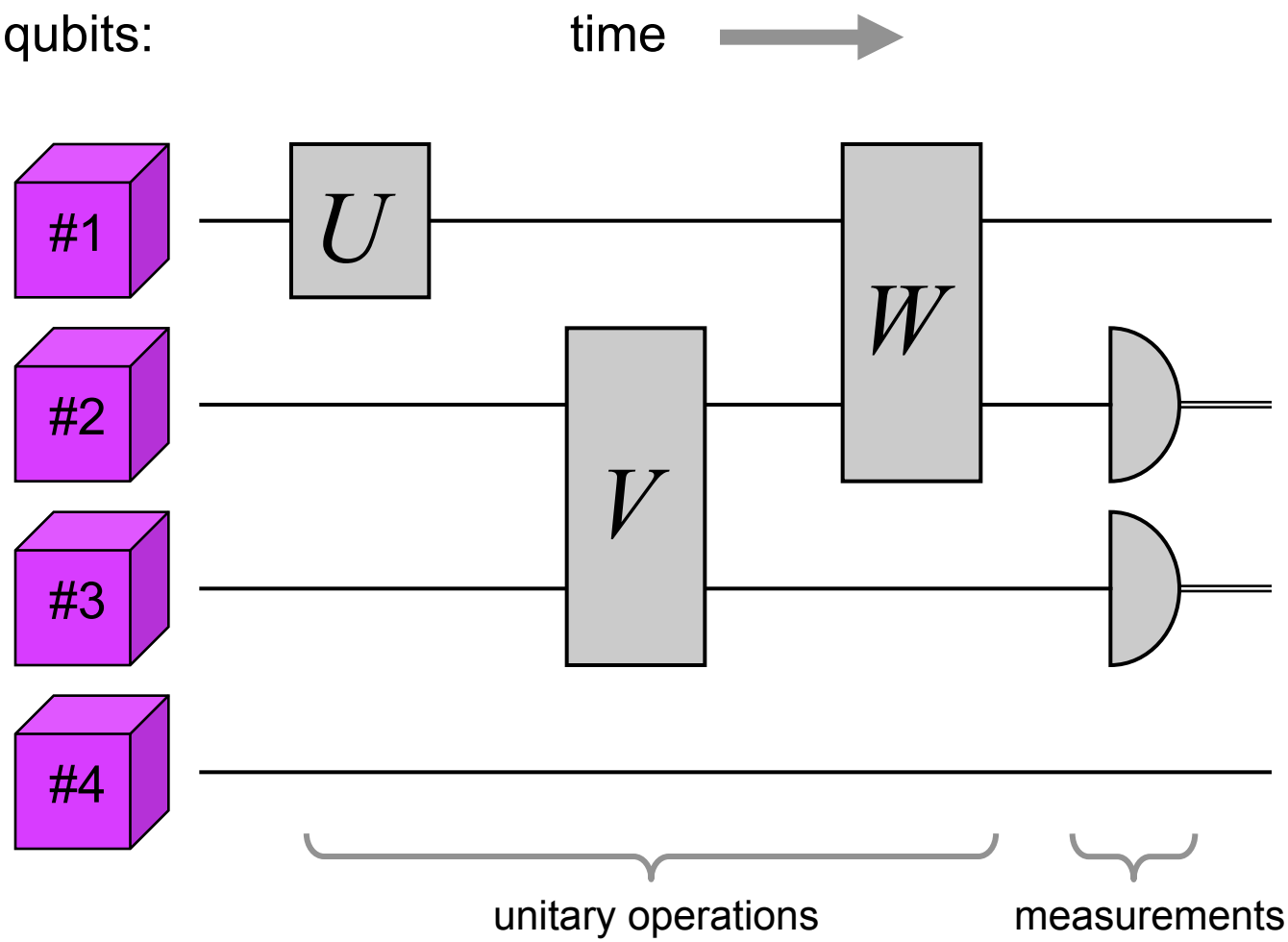
Controlled-NOT (CNOT)



Note: "control" qubit may change on some input states!

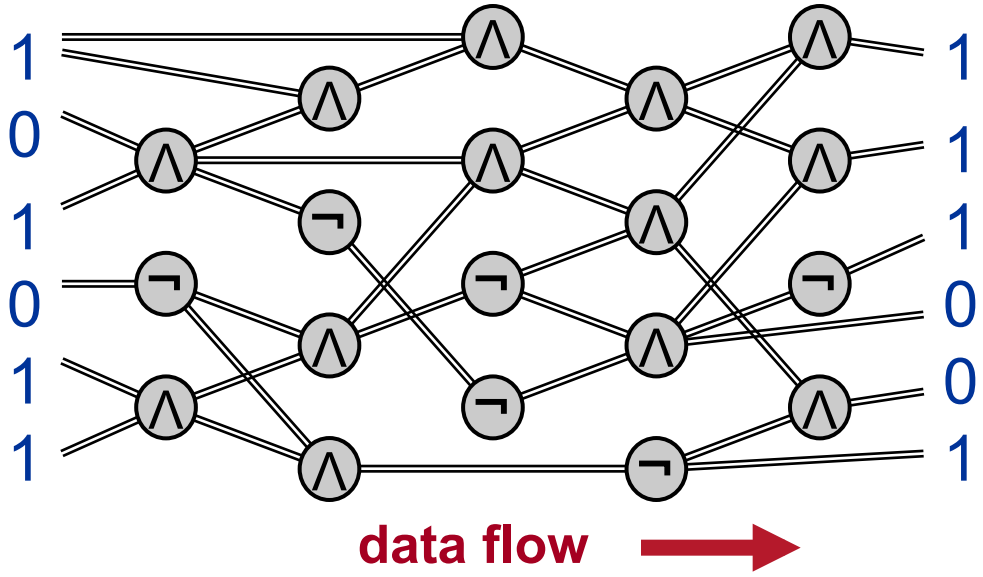


Structure among subsystems

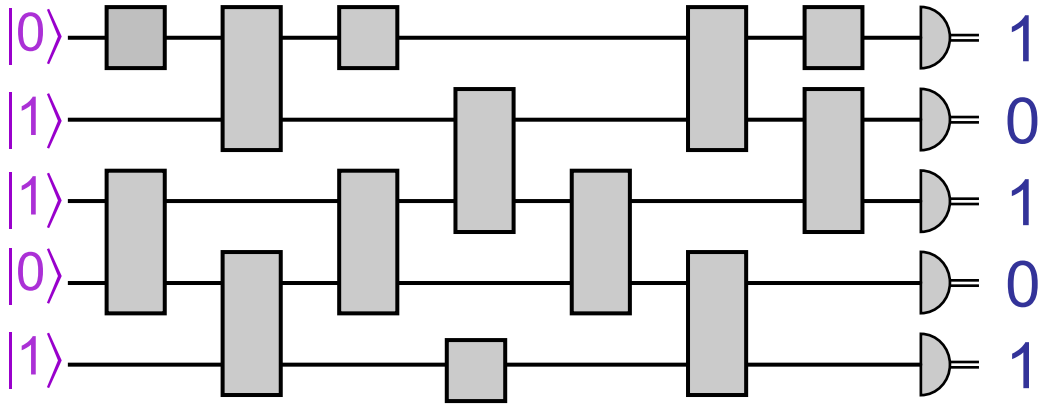


Models of computation

Classical circuits:



Quantum circuits:

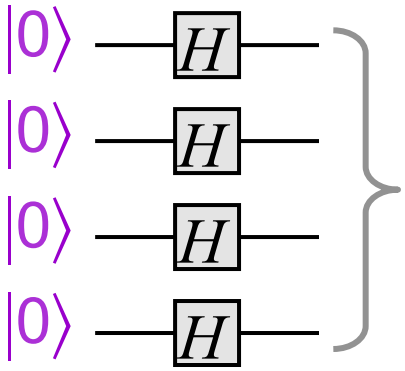


Circuit families of size $O(n^2 \operatorname{polylog} n)$ exist for prime factorization, whereas the best known classical circuit family size is $\approx 2^{n^{1/3}}$

Equally weighted superpositions

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

$$H|0\rangle = \tfrac{1}{\sqrt{2}}|0\rangle + \tfrac{1}{\sqrt{2}}|1\rangle$$



$$H \otimes H \otimes \ldots \otimes H|00\ldots0\rangle$$

$$= \left(\tfrac{1}{\sqrt{2}}|0\rangle + \tfrac{1}{\sqrt{2}}|1\rangle\right) \otimes \left(\tfrac{1}{\sqrt{2}}|0\rangle + \tfrac{1}{\sqrt{2}}|1\rangle\right) \otimes \cdots \otimes \left(\tfrac{1}{\sqrt{2}}|0\rangle + \tfrac{1}{\sqrt{2}}|1\rangle\right)$$

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

How do quantum algorithms work?

Given a polynomial-time classical algorithm for $f:\{0,1\}^n \rightarrow T$, it is straightforward to construct a quantum algorithm that creates the state

$$\frac{1}{\sqrt{2^n}} \sum_x |x\rangle |f(x)\rangle$$

at the cost of about ***one*** evaluation of f

Is this exponentially many computations at polynomial cost?

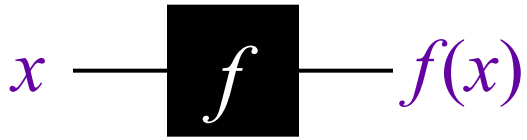
No! — the most straightforward way of extracting information from the state yields just $(x, f(x))$ for a random $x \in \{0,1\}^n$

But we can make some interesting ***tradeoffs***:

instead of learning about any $(x, f(x))$ point, one can learn something about a ***global property*** of f

Query scenario

Input: a function f, given as a black box (a.k.a. oracle)



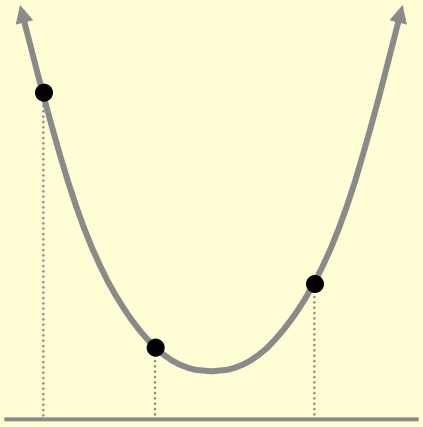
Goal: determine some information about f making as few queries to f (and other operations) as possible

Example: polynomial interpolation

Let: $f(x) = c_0 + c_1 x + c_2 x^2 + \ldots + c_d x^d$

Goal: determine $c_0, c_1, c_2, \ldots, c_d$

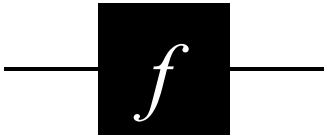
Question: How many f-queries does one require for this?



Answer: $d+1$

Example 1: Deutsch's problem

Let $f: \{0,1\} \rightarrow \{0,1\}$



There are ***four*** possibilities:

x	$f_1(x)$
0	0
1	0

x	$f_2(x)$
0	1
1	1

x	$f_3(x)$
0	0
1	1

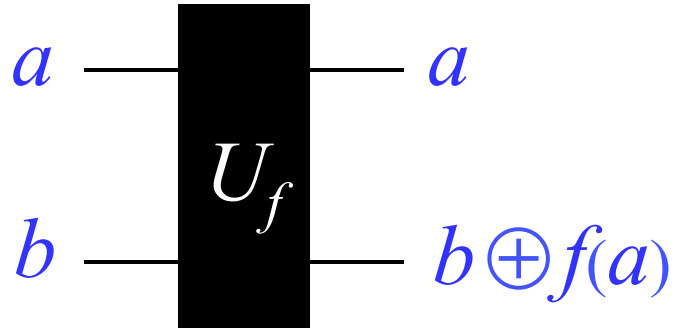
x	$f_4(x)$
0	1
1	0

Goal: determine whether or not $f(0) = f(1)$ (i.e. $f(0) \oplus f(1)$)

Any classical method must make ***two*** queries

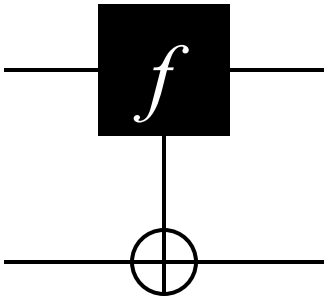
There is a quantum algorithm that makes only ***one*** query

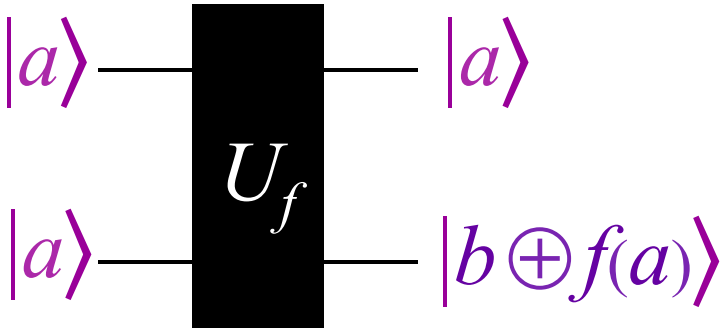
Reversible black box for f



input output

alternate notation:





input output

Example 2: search problem

Given: a black box computing $f: \{0,1\}^n \rightarrow \{0,1\}$

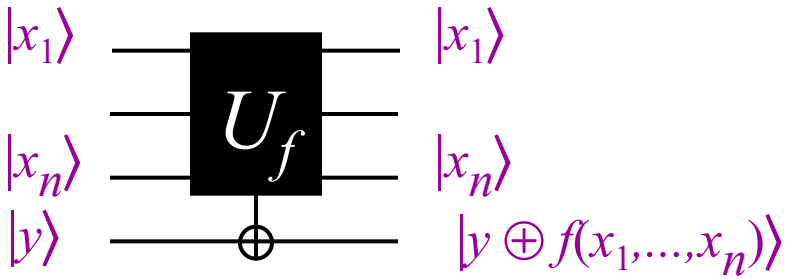
Goal: determine if f is ***satisfiable*** (if $\exists x \in \{0,1\}^n$ s.t. $f(x) = 1$)

In positive instances, it makes sense to also ***find*** such a satisfying assignment x

Classically, using probabilistic procedures, order 2^n queries are necessary to succeed—even with probability ¾ (say)

Grover's ***quantum*** algorithm that makes only $O(\sqrt{2^n})$ queries

Query:

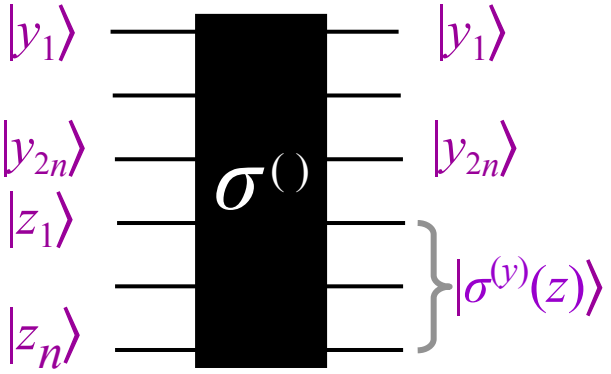


[Grover '96]

Example 3: period finding

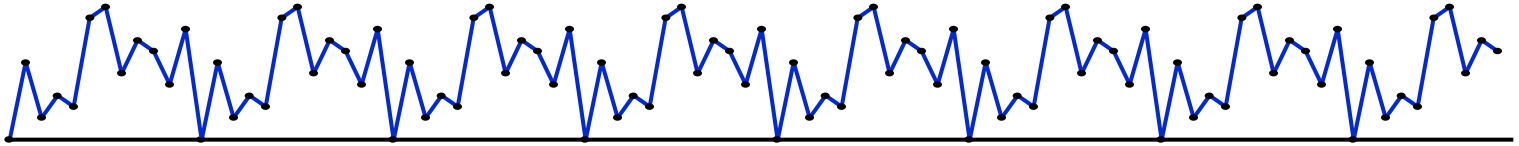
Let $\sigma : \{0,1\}^n \rightarrow \{0,1\}^n$ be a(n unknown) permutation, and let $\sigma^{(y)}(z)$ denote y iterations of σ, namely $\sigma(\sigma(\sigma(\ldots\ \sigma(z)\ \ldots)))$

Given: a black box computing $\sigma^{(y)}(z)$, where $y \in \{0,1\}^{2n}$ and $z \in \{0,1\}^n$



(enables fast forwarding of large iterations of σ)

Goal: determine length of cycle containing 0 (say)



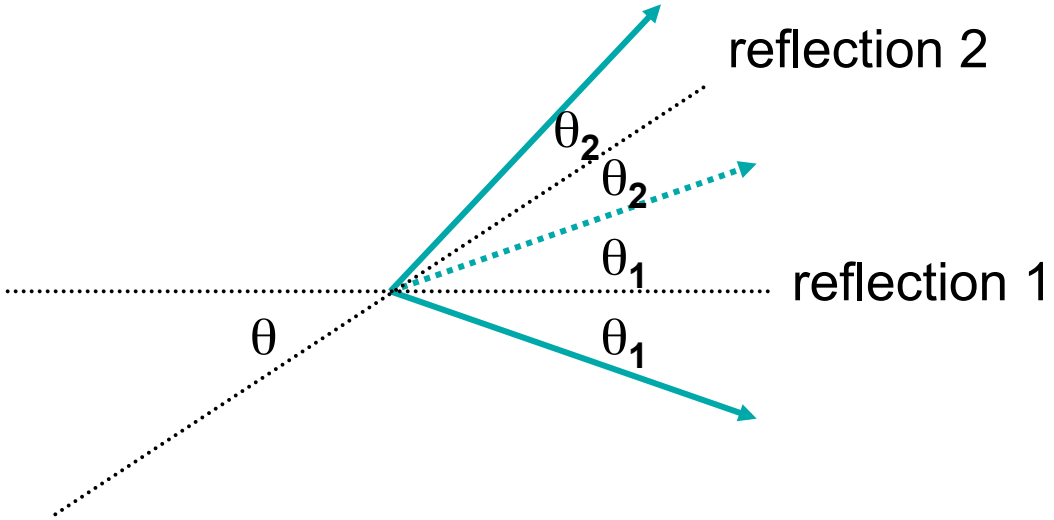
$$f(y) = \sigma^{(y)}(00\ldots0)$$

Query complexity: classically: $\Omega(2^{n/3})$ queries; quantumly: $O(1)$

Prelude to Grover's search algorithm:

two reflections = a rotation

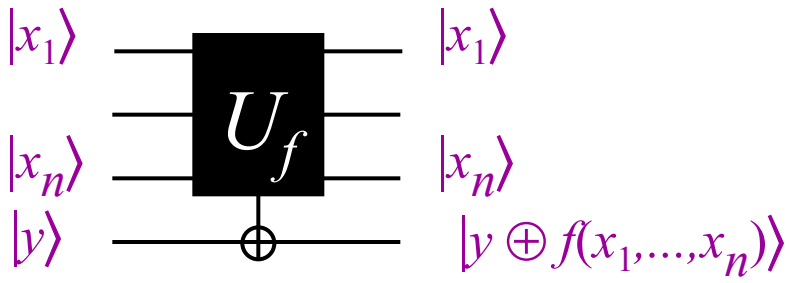
Consider two lines with intersection angle θ:



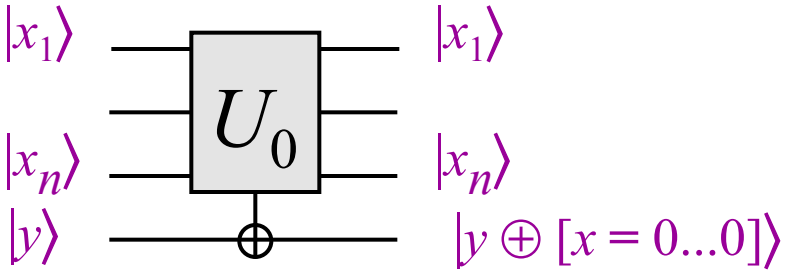
Net effect: rotation by angle 2θ, ***regardless of starting vector***

Grover's algorithm: description I

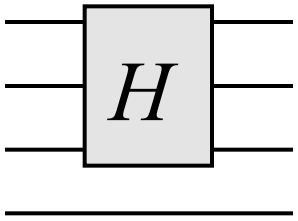
Basic operations used:



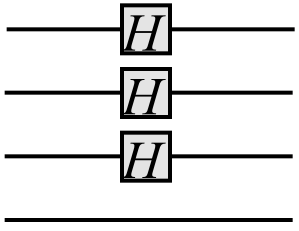
$$U_f |x\rangle(|0\rangle-|1\rangle) = (-1)^{f(x)} |x\rangle(|0\rangle-|1\rangle)$$



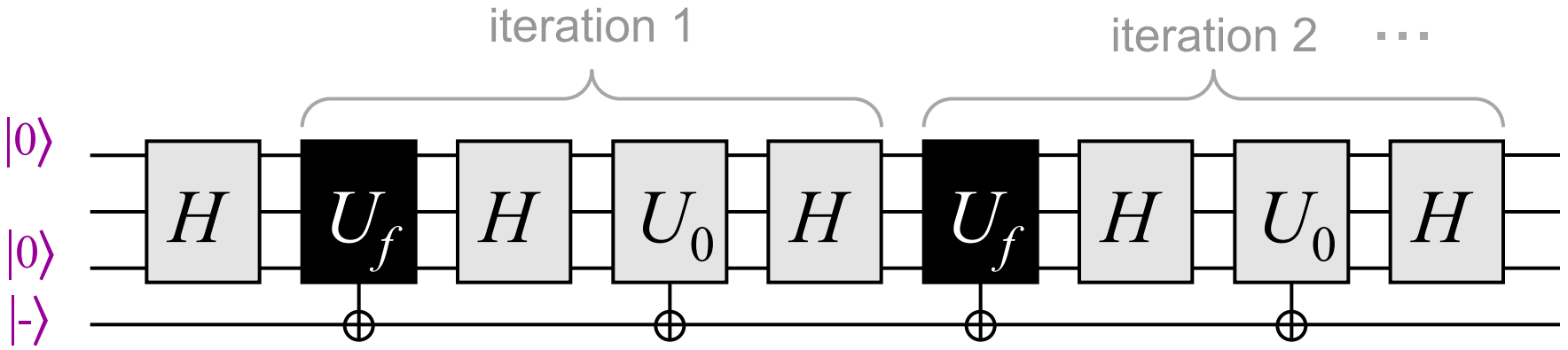
$$U_0 |x\rangle(|0\rangle-|1\rangle) = (-1)^{[x\,=\,0\ldots0]} |x\rangle(|0\rangle-|1\rangle)$$



Hadamard



Grover's algorithm: description II



1. construct state $H|0...0\rangle(|0\rangle-|1\rangle)$
2. repeat k times:

   apply $-HU_0HU_f$ to state
3. measure state, to get $x\in\{0,1\}^n$, and check if $f(x)=1$

(The setting of k will be determined later)

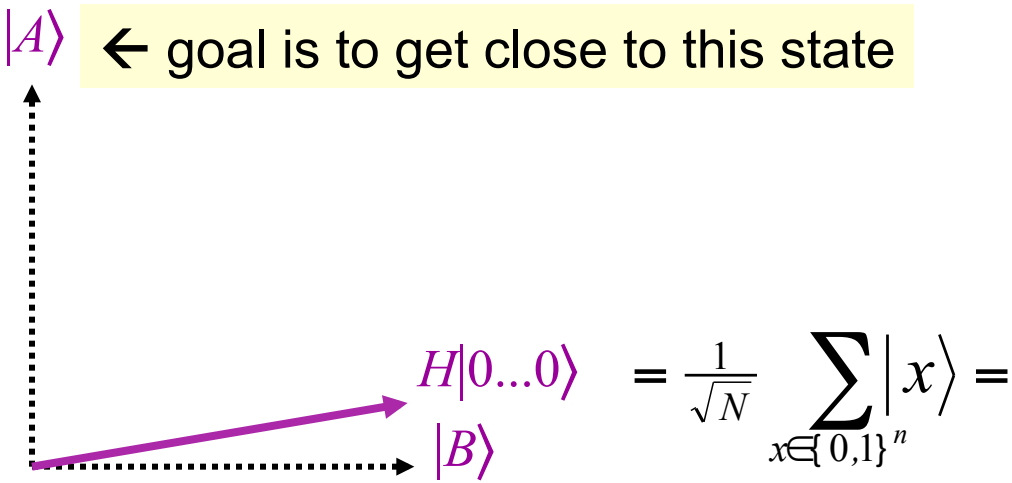
Grover's algorithm: analysis I

Let $A = \{x \in \{0,1\}^n : f(x) = 1\}$ and $B = \{x \in \{0,1\}^n : f(x) = 0\}$

and $N = 2^n$ and $a = |A|$ and $b = |B|$

Let $|A\rangle = \frac{1}{\sqrt{a}} \sum_{x \in A} |x\rangle$ and $|B\rangle = \frac{1}{\sqrt{b}} \sum_{x \in B} |x\rangle$

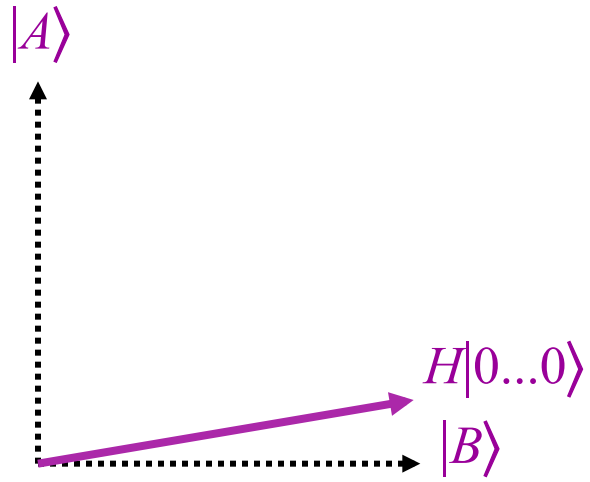
Consider the space spanned by $|A\rangle$ and $|B\rangle$



$$= \frac{1}{\sqrt{N}} \sum_{x \in \{0,1\}^n} |x\rangle = \sqrt{\frac{a}{N}} |A\rangle + \sqrt{\frac{b}{N}} |B\rangle$$

Interesting case: $a << N$

Grover's algorithm: analysis II



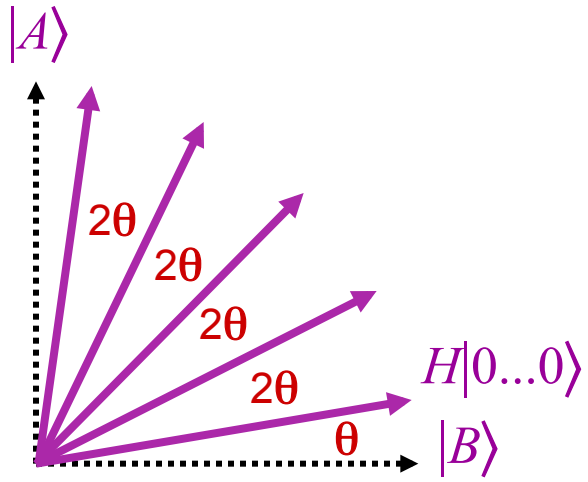
Algorithm: $\left(-HU_0HU_f\right)^k H|0\ldots0\rangle$

Observations:

U_f is a reflection about $|B\rangle$: $U_f|A\rangle = -|A\rangle$ and $U_f|B\rangle = |B\rangle$

$-HU_0H$ is a reflection about $H|0\ldots0\rangle$

Grover's algorithm: analysis III



Algorithm: $(-HU_0HU_f)^k H|0...0\rangle$

Since $-HU_0HU_f$ is a composition of two reflections, it is a rotation by **2θ**, where $\sin(\theta)=\sqrt{a/N} \approx \sqrt{a/N}$

When $a = 1$, we want $(2k+1)(1/\sqrt{N}) \approx \pi/2$, so $k \approx (\pi/4)\sqrt{N}$

More generally, it suffices to set $k \approx (\pi/4)\sqrt{N/a}$

(Can be adapted to work even if a is not known in advance)

Applications of quantum search

The function f could be realized as a **3-CNF formula**:

$$f(x_1,\ldots,x_n) = (x_1 \vee \bar{x}_3 \vee x_4) \wedge (\bar{x}_2 \vee x_3 \vee \bar{x}_5) \wedge \cdots \wedge (\bar{x}_1 \vee x_5 \vee \bar{x}_n)$$

Running time: $O(\sqrt{2^n}\,\mathrm{polylog}\,n)$ gates

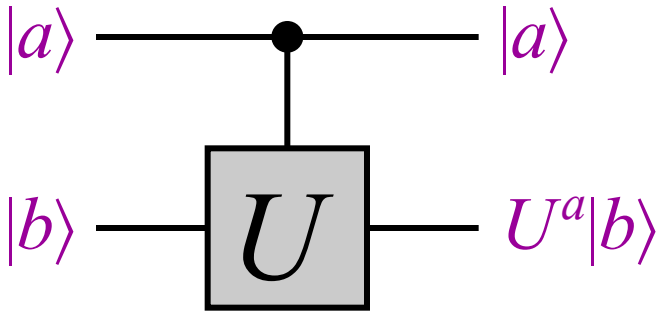
But there are better ***classical*** SAT solvers, such as [Schöning '99]:

$O(\sqrt{2^n}\,\mathrm{polylog}\,n)$ gates

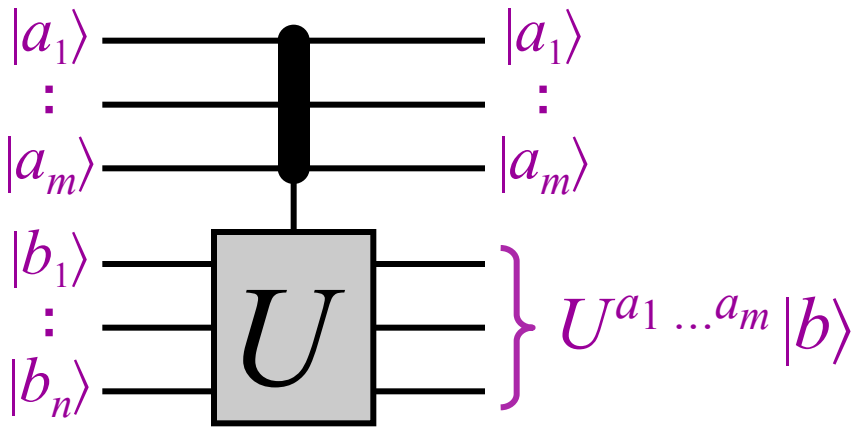
We can recast Schöning's algorithm as a search on a space of size $O(\sqrt{2^n}\,\mathrm{polylog}\,n)$ and then use Grover's algorithm to attain algorithm with $O((2^n)^{\mathbf{1/4}}\,\mathrm{polylog}\,n)$ gates

That's essentially what is called ***amplitude amplification***

Generalized controlled-U gates



$$\begin{bmatrix} I & 0 \\ 0 & U \end{bmatrix}$$



$$\begin{bmatrix} I & 0 & 0 & \cdots & 0 \\ 0 & U & 0 & \cdots & 0 \\ 0 & 0 & U^2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & U^{2^m-1} \end{bmatrix}$$

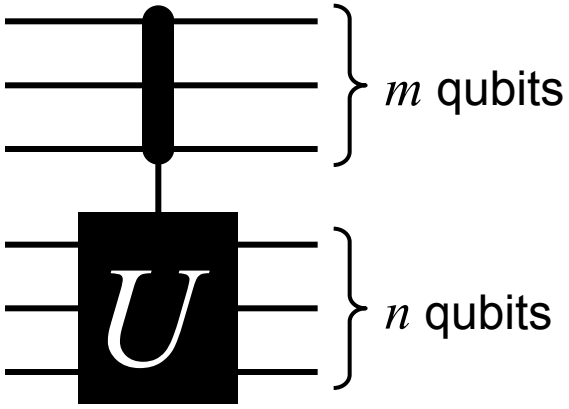
Example: $|1101\rangle|0101\rangle \rightarrow |1101\rangle U^{1101}|0101\rangle$

Eigenvalue estimation problem

U is a unitary operation on n qubits

$|\psi\rangle$ is an eigenvector of U, with eigenvalue $e^{2\pi i\phi}$
$(0 \leq \phi < 1)$

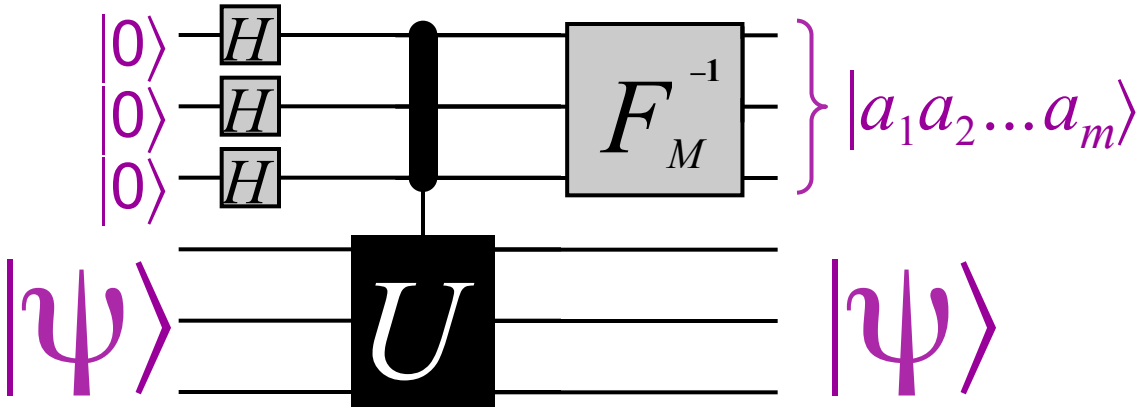
Input: black-box for



and a copy of $|\psi\rangle$

Output: ϕ (m-bit approximation)

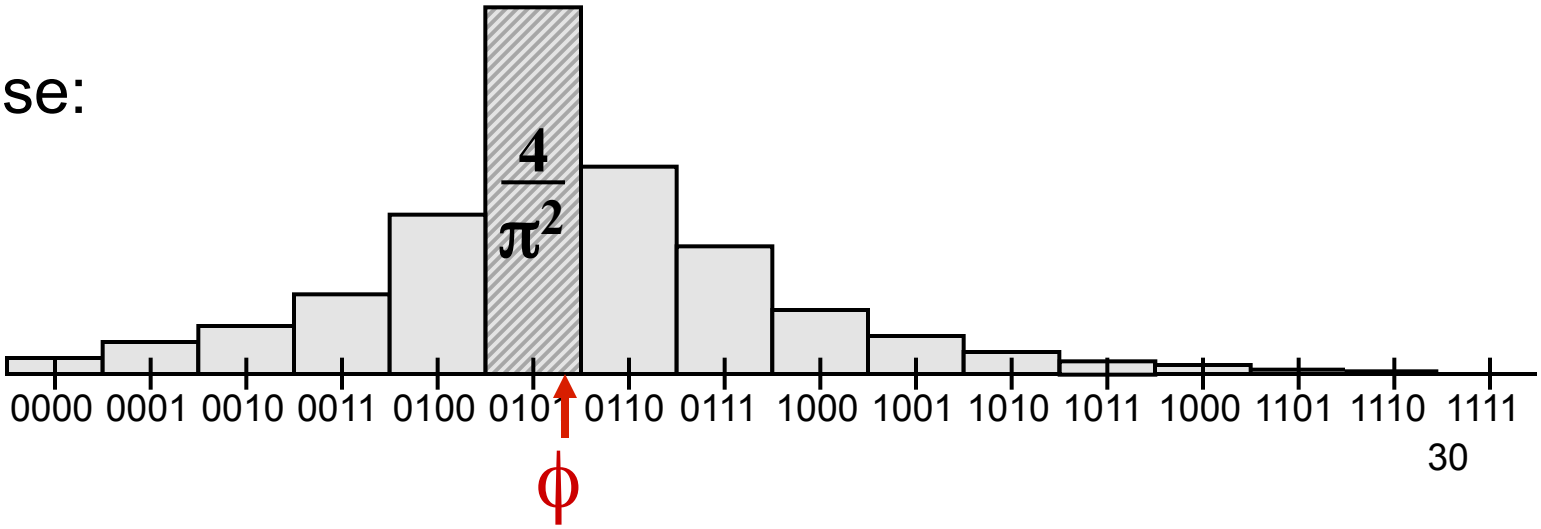
Algorithm for eigenvalue estimation



F_M is quantum Fourier transform

If $\phi = 0.a_1 a_2 \ldots a_m$ then the above procedure yields $|a_1 a_2 \ldots a_m\rangle$ (from which ϕ can be deduced exactly)

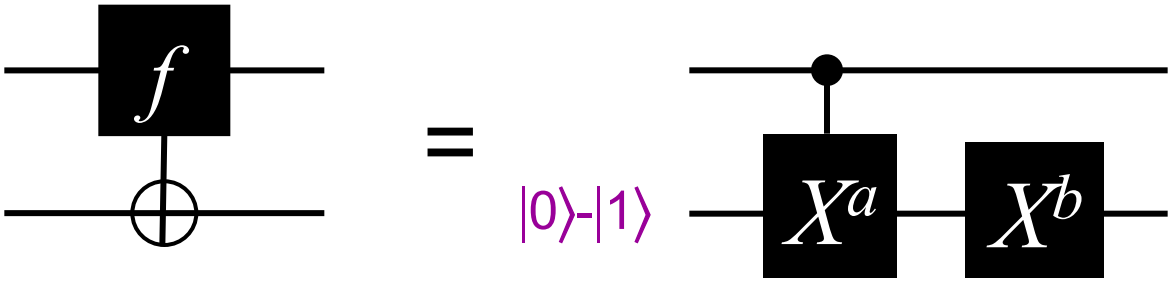
Otherwise:



Simple example: Deutsch's problem

Recall $f: \{0,1\} \rightarrow \{0,1\}$ $\qquad f(z) = a \cdot z + b \bmod 2$

Note that f is constant iff $a = 0$



global phase that can be ignored

Eigenvector of X^a is $|\psi\rangle = |0\rangle - |1\rangle$

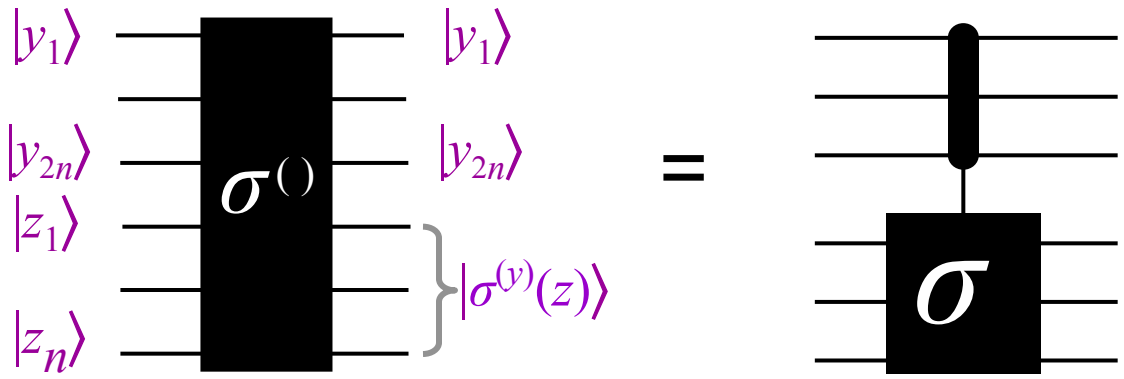
Eigenvalue of X^a is: **+1** if constant; **-1** if balanced

Eigenvalue can be determined with a single query to f

Another example: period finding

Recall that $\sigma : \{0,1\}^n \rightarrow \{0,1\}^n$ is an unknown permutation, and $\sigma^{(y)}(z)$ denotes y iterations of σ, namely $\sigma(\sigma(\sigma(\ldots\ \sigma(z)\ \ldots)))$

Eigenvectors of σ: $|\psi_1\rangle = \frac{1}{\sqrt{r}} \sum_{j=0}^{r} |\sigma^{(j)}(0)\rangle \qquad |\psi_k\rangle = \frac{1}{\sqrt{r}} \sum_{j=0}^{r} e^{-i2k\pi/r} |\sigma^{(j)}(0)\rangle$



Eigenvalue is k/r from which r can be deduced by continued fractions

Trick for bypassing construction of $|\psi_k\rangle$ by using $|0\rangle$

Order-finding (min r such that $a^r \bmod m = 1$) reduces to **period finding** (fast forward $\sigma(z) = az \bmod m$ by repeated squaring)

Thank you