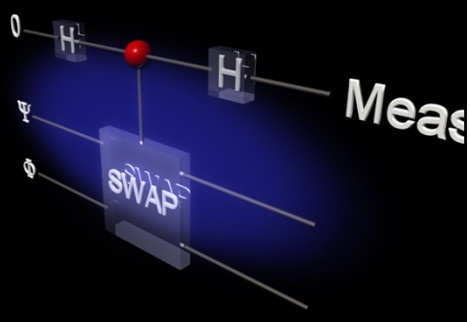
Introduction to Quantum Algorithms

11th Canadian Summer School on Quantum Information

Michele Mosca mmosca@iqc.ca

Canada Research Chair in Quantum Computation



IQC Institute for Quantum Computing

UNIVERSITY OF WATERLOO

PI

Computational Complexity

- *The (computational) complexity* of an **algorithm** refers to some measure of the resources *(e.g. time, space, basic operations, energy)* used by the algorithm.
- E.g. the traditional algorithm for multiplying two n-bit numbers takes $O(n^2)$ **time** steps (with pen and paper, or on a PC).
- E.g. the best-known rigorous probabilistic classical algorithms for factoring an n-digit number into its prime factors with high probability takes **time** in

$$e^{O\left(\sqrt{n \log n}\right)}.$$

- We measure the complexity as a function of the input size.
- Unless stated otherwise, we refer to the "worst-case" complexity, i.e. the running time on a worst-case input.

2

Computational Complexity

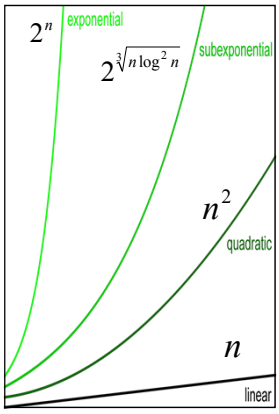
- *The (computational) complexity of a* ***problem*** refers to some measure of the resources (e.g. *time, space, basic operations, energy*) required to solve a problem.
- E.g. multiplying two *n*-bit numbers must take $\Omega(n)$ **time** (even just to write the answer)**.**
- No known algorithm achieves that lower bound.
  - Until 2007, the best known upper bound was (Schönhage-Strassen, 1971) $O(n \log n \log\log n)$
  - In 2007, this was improved (Fürer) to $O(n \log n 2^{\log^* n})$
- Unless stated otherwise, we refer to the "worst-case" complexity, i.e. the resources required to solve the problem on any input of size *n*.

3

Aside : Computationally secure cryptography

Why do we believe the current cryptographic tools are secure?

- It is easy to multiply two large numbers 9287137268623 **x** 61649017818402059 = 572542890955285156447699294757
- It takes between *n* and n^2 steps to multiply two *n*-digit numbers.
- It is not known how to easily factor an arbitrary large (non-prime) number (e.g. 652186080857407065896971) into a product of smaller numbers.
- Best-known heuristic classical methods take roughly $2^{\sqrt[3]{n \log^2 n}}$ steps.
- The best known classical methods for breaking elliptic curve cryptography with *n* digit keys take roughly 2^n steps.



4

Really big numbers …

A million GHz computers, running for 100 years, can perform about 2^{71} basic operations

10^6 computers x 10^9 cycles/second/computer x 31536000 seconds/year x 100 years
$\approx 2^{20} \times 2^{30} \times 2^{24} \times 2^7 = 2^{71}$ cycles

A billion THz computer, running for 1000 years, can perform about 2^{104} basic operations

10^9 computers x 10^{12} cycles/second/computer x 31536000 seconds/year x 10^3 years
$\approx 2^{30} \times 2^{40} \times 2^{24} \times 2^{10} = 2^{104}$ cycles

Approximate age of the universe: 2^{92} seconds

5

Big numbers …

www.keylength.com

The best known classical algorithm for cracking 256-bit Elliptic Curve Crypto requires about 2^{128} steps

The best known classical algorithm for cracking 2030-bit RSA requires about 2^{128} steps

The best known classical algorithm for cracking 128-bit AES (a widely used symmetric key cipher) requires about 2^{128} steps

"Polynomial" cost

- We say an algorithm A runs in "polynomial" time if there exists a polynomial p(n) such that the algorithm takes time at most p(n) on inputs of size n.
- One can similarly talk about algorithms that use polynomial space, or a polynomial number of logic gates.
- Algorithm A simulates algorithm B with "polynomial overhead" (in time e.g.) if there is some polynomial p(n) such that when algorithm B uses T time then algorithm A takes time at most p(T) when simulating algorithm B. (similarly for other resources)

7

Why polynomials ??

- Polynomials are closed under addition, multiplication and composition.
- Any known realistic classical computing model can be simulated with polynomial overhead by a classical (probabilistic) Turing machine (or a PC with arbitrary memory and random coins). Thus, in these cases, the Strong Church-Turing thesis holds.
- In general, measuring complexity up to a polynomial factor gives a certain degree of robustness (e.g. against reasonable changes in computing model, and other "details")

8

Polynomial complexity ≈ Efficient

"It should not come as a surprise that our choice of polynomial algorithms as the mathematical concept that is supposed to capture the informal notion of 'practically efficient computation' is open to criticism from all sides. […]

Ultimately, our argument for our choice must be this: **Adopting polynomial worst-case performance as our criterion of efficiency results in an elegant and useful theory that says something meaningful about practical computation, and would be impossible without this simplification**" – Christos Papadimitriou

9

How does quantum information change computational complexity?

- We don't think a classical computer efficiently captures the computational power of a quantum universe.

- Quantum Strong Church-Turing thesis:
  Any known realistic computing model can be simulated with polynomial overhead by a *quantum* computer.

Different computational models

Increasing (?) capabilities

Increasing (?) capabilities

	Closed system (i.e. reversible)	**Open system (i.e. not necessarily reversible)**	
classical	Classical reversible circuit model	(without randomness) Deterministic classical circuit model	(with randomness) Probabilistic classical circuit model
quantum	Quantum circuit model with unitary gates	Quantum circuit model with general quantum gates	

11

Universal set of quantum gates

To capture the full computational power of quantum information (as defined in previous lectures), it suffices to have a *universal set of unitary quantum gates*.

Definition

A set of gates *G* is said to be universal if for any integer $n>0$, any n-qubit unitary operator can be approximated to arbitrary accuracy by a quantum circuit using only gates from *G*.

Results about universality give us guidelines for useful implementation paradigms, as well as for useful algorithmic paradigms.

12

Definition of *error* or *accuracy*

Suppose we approximate a desired unitary transformation *U* by some other unitary transformation *V*.

The *error* in the approximation is defined to be

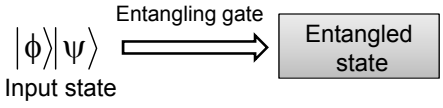
$$E(U,V) \equiv \max_{|\psi\rangle} || U|\psi\rangle - V|\psi\rangle ||$$

13

Universal set of quantum gates

Definition

A two-qubit gate is said to be entangling if for some input product state, the output of the gate is an entangled state.



14

Universal set of quantum gates

Theorem:
A set composed of any two-qubit entangling gate, together with all one-qubit gates, is universal.

… a bit of an overkill, since such a set allows one to achieve any unitary exactly.

Also unrealistic, since one needs access to an infinite number of one-qubit gates.

Can we achieve universality with a finite set of gates?

15

Arbitrary one-qubit operations

Theorem (one-qubit universality):
Let $\hat{n}$ and $\hat{m}$ be any two non-parallel axes of the Bloch sphere, and let β, γ be real numbers such that

$$\frac{\beta}{\pi}, \frac{\gamma}{\pi}$$

are not rational.

Then

$$G = \left\{R_{\hat{n}}(\beta), R_{\hat{m}}(\gamma)\right\}$$

is universal for one-qubit gates.

16

A universal set of gates: an example

Theorem:
The set

$$G = \{H, T, CNOT\}$$

is a universal set of gates.

i.e. any *n*-qubit unitary operator *U* can be approximated with error ε, for any $\varepsilon > 0$, using a finite circuit with gates from *G*.

17

Efficiency of approximation

How does the size of a circuit scale as the desired accuracy improves?

$e^{O\left(\frac{1}{\varepsilon}\right)}$? $\quad O\left(\frac{1}{\varepsilon}\right)$? $\quad O\left(\log\frac{1}{\varepsilon}\right)$?

18

Solovay-Kitaev theorem

Theorem:

If *G* is a finite set of one-qubit gates satisfying the conditions of the one-qubit universality theorem, and

iii) for any gate $g \in G$, its inverse g^{-1} can be implemented exactly by a finite sequence of gates in *G*

➡ any one-qubit gate can be approximated with error at most ε using $O\left(\log^c(1/\varepsilon)\right)$ gates from *G*, where *c* is a positive constant.

19

Efficiency of approximation

Corollary

It is possible to approximate a circuit with *T* gates from any universal set with $O\left(T \log^c(T/\varepsilon)\right)$ gates from any finite universal set of gates satisfying condition *iii)*.

- Key points are sketched in section 4.4 of KLM textbook.
- More details in Appendix of N&C, or KSV.

20

Bottom line for computer algorithmics

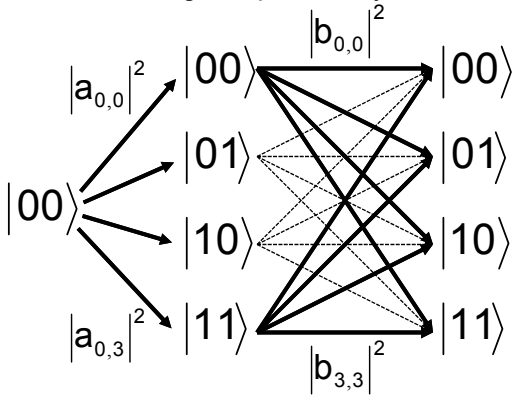
It suffices to design algorithms with your favourite finite universal gate set.

However, one might gain intuition or other practical advantages by working in other equivalent algorithmic paradigms (e.g. quantum walks, adiabatic algorithms, measurement-based, topological, etc)

21

A classical randomized algorithm

The probabilities could correspond to the square of a probability amplitude (due to measuring the quantum system at each time step)

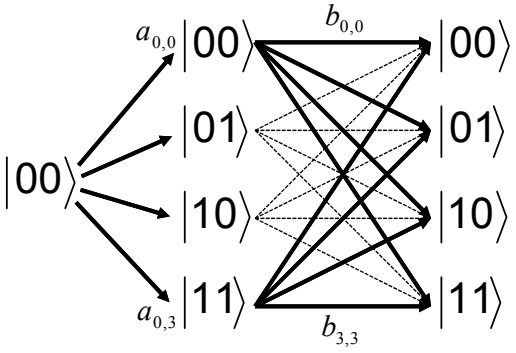


$$\Pr(00) = \sum_j \left|a_{0,j} b_{j,0}\right|^2$$

22

A quantum algorithm

If we don't measure at each time step, only at the end, the probability amplitudes first have a chance to interfere.



$$\Pr(00) = \left| \sum_j a_{0,j} b_{j,0} \right|^2$$

23

Decoherence

- A quantum system that is continually measured (or "leaks" information to an external system) will behave like a classical randomized system.
- Partial measurements will give a probability distribution somewhere in between the two extremes.
- Error-correcting codes will allow a quantum system interacting with the environment to maintain "coherence".

24

How do quantum algorithms work?

Given a polynomial-time classical algorithm for $f:\{0,1\}^n \to T$, 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

25

Quantum algorithms

- Quantum Algorithms should exploit quantum parallelism *and* quantum interference.
- This is necessary, but not sufficient, in order to outperform a classical probabilistic algorithm.

  E.g. at some point in the execution of the algorithm, the state of the system should have a substantial amount of entanglement (assuming we are in the usual model of unitary operations on pure states).

26

Query scenario

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

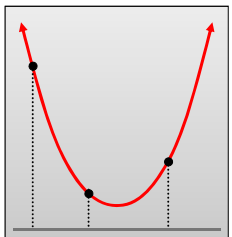
Goal: determine some information about f making as few queries to f as possible (of course, other operations are allowed – but we do not count them)

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 classical f-queries does one require for this?



Answer: $d+1$

27

- Introduction to quantum algorithms
- **Parity problem and Deutsch's algorithm**
- Constant vs. balanced problem
- Computing H⊗H⊗ ... ⊗H
- Simon's problem

28

Deutsch's problem

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

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

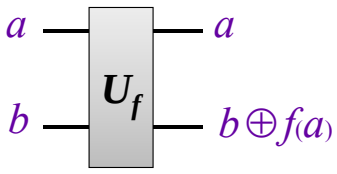
Goal: determine whether or not $f(0) = f(1)$ (i.e. $f(0) \oplus f(1)$)

Any classical method requires ***two*** queries

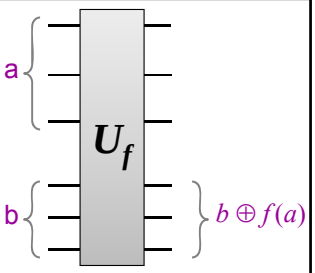
What about a quantum method?

29

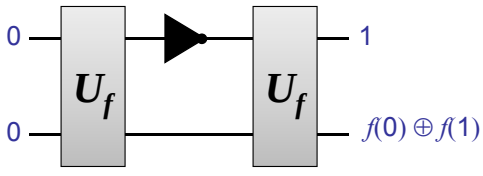
Unitary black box for *f*



a and b can be more than one qubit



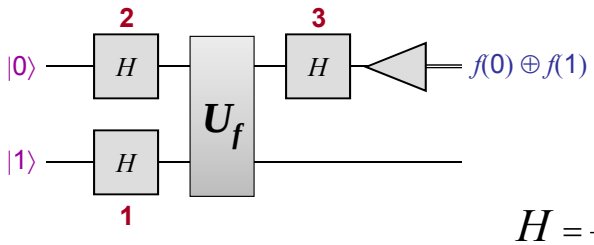
A classical algorithm:
(still requires 2 queries)



2 queries + **1** auxiliary operation

30

Quantum algorithm for Deutsch



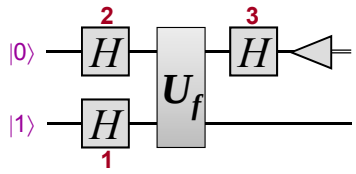
$$H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

1 query + **4** auxiliary operations

How does this algorithm work?

Each of the three H operations can be seen as playing a different role ...

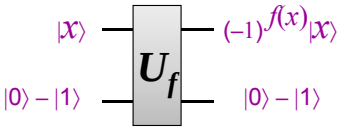
Quantum algorithm (1)



1. Creates the state $|0\rangle - |1\rangle$, which is an eigenvector of

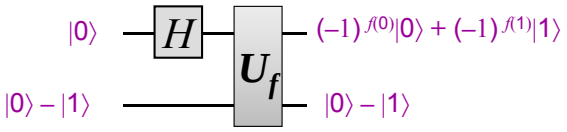
- **NOT** with eigenvalue -1
- $\boldsymbol{I}$ with eigenvalue $+1$

This causes f to induce a ***phase shift*** of $(-1)^{f(x)}$ to $|x\rangle$



Quantum algorithm (2)

2. Causes f to be queried ***in superposition*** (at $|0\rangle + |1\rangle$)



x	$f_1(x)$
0	0
1	0

x	$f_2(x)$
0	1
1	1

$\pm(|0\rangle + |1\rangle)$

x	$f_3(x)$
0	0
1	1

x	$f_4(x)$
0	1
1	0

$\pm(|0\rangle - |1\rangle)$

Quantum algorithm (3)

3. Distinguishes between $\pm(|0\rangle + |1\rangle)$ and $\pm(|0\rangle - |1\rangle)$

$$\pm(|0\rangle + |1\rangle) \xleftrightarrow{H} \pm|0\rangle$$

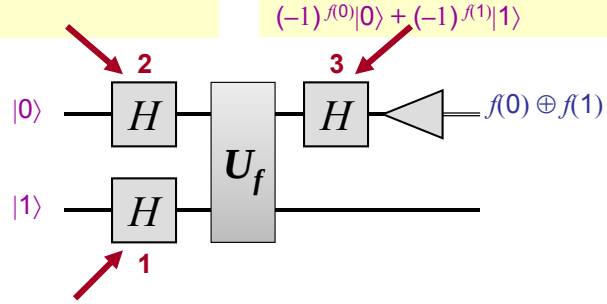
$$\pm(|0\rangle - |1\rangle) \xleftrightarrow{H} \pm|1\rangle$$

Summary of Deutsch's algorithm

Makes only one query, whereas two are needed classically

produces superpositions of inputs to f: $|0\rangle + |1\rangle$

extracts phase differences from $(-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle$



constructs eigenvector so f-queries induce phases: $|x\rangle \rightarrow (-1)^{f(x)}|x\rangle$

- Introduction to quantum algorithms
- Parity problem and Deutsch's algorithm
- **Constant vs. balanced problem**
- Computing H⊗H⊗ ... ⊗H
- Simon's problem

Constant vs. balanced

Let $f: \{0,1\}^n \rightarrow \{0,1\}$ be either constant or balanced, where

- ***constant*** means $f(x) = 0$ for all x, or $f(x) = 1$ for all x
- ***balanced*** means $\Sigma_x\ f(x) = 2^{n-1}$

Goal: determine whether f is constant or balanced

How many queries are there needed *classically?* $2^{n-1}+1$

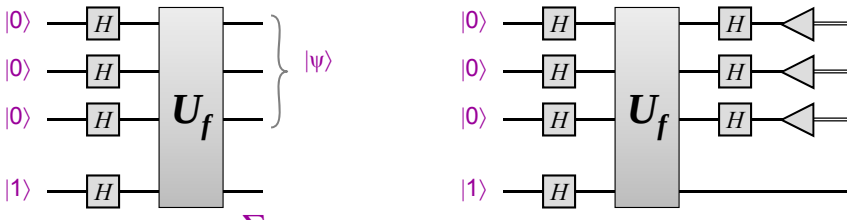
Example: if $f(0000) = f(0001) = f(0010) = ... = f(0111) = 0$ then it still could be either

Quantumly? just 1 query suffices!

[Deutsch & Jozsa, 1992]

37

Quantum algorithm



Constant case: $|\psi\rangle = \pm \Sigma_x |x\rangle$ ***Why?***

Balanced case: $|\psi\rangle$ is ***orthogonal*** to $\pm \Sigma_x |x\rangle$ ***Why?***

How to distinguish between the cases? What is $H^{\otimes n}|\psi\rangle$?

Constant case: $H^{\otimes n}|\psi\rangle = \pm |00...0\rangle$

Balanced case: $H^{\otimes n}|\psi\rangle$ is orthogonal to $|0...00\rangle$

Last step of the algorithm: if the measured result is 000 then output "constant", otherwise output "balanced"

38

Probabilistic *classical* algorithm solving constant vs. balanced

But here's a classical procedure that makes only **2** queries and performs fairly well probabilistically:

1. pick $x_1, x_2 \in \{0,1\}^n$ randomly
2. **<u>if</u>** $f(x_1) \neq f(x_2)$ **<u>then</u>** output balanced **<u>else</u>** output constant

What happens if f is constant? The algorithm always succeeds

What happens if f is balanced? Succeeds with probability ½

By repeating the above procedure k times:
$2k$ queries and one-sided error probability $(½)^k$

Therefore, for "bounded-error" algorithms, the improvement doesn't scale as a function of n, though there is an improvement with respect to error probability.

39

- Introduction to quantum algorithms
- Parity problem and Deutsch's algorithm
- Constant vs. balanced problem
- **Computing H⊗H⊗ ... ⊗H**
- Simon's problem

40

Recap

- Quantum Computational Complexity is an elegant robust framework for studying the efficient computability of computational problems
  - A guiding tool for computationally secure cryptography
  - Notion of polynomial cost and polynomial overhead is critical
  - Quantum Strong Turing thesis
  - Efficient universal quantum computation
- Quantum Algorithms exploit quantum parallelism and quantum interference to solve problems more efficiently than the best-known classical algorithms
  - Black-box (or "query") complexity is a useful paradigm for studying quantum algorithms and complexity
  - Deutsch's "constant vs balanced" problem and solution gave us the first quantum algorithm; several developments eventually led to Simon's algorithm and then Shor's algorithms

41

About $H \otimes H \otimes \ldots \otimes H = H^{\otimes n}$

Theorem: for $x \in \{0,1\}^n$, $H^{\otimes n}|x\rangle = \frac{1}{2^{n/2}} \sum_{y \in \{0,1\}^n} (-1)^{x \cdot y} |y\rangle$

where $x \cdot y = x_1 y_1 \oplus \ldots \oplus x_n y_n$

Example: $H \otimes H = \frac{1}{2} \begin{bmatrix} +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 \end{bmatrix}$

Proof: For all $x \in \{0,1\}$, $H|x\rangle = |0\rangle + (-1)^x |1\rangle = \Sigma_y (-1)^{xy} |y\rangle$

Thus, $H^{\otimes n}|x_1 \ldots x_n\rangle = (\Sigma_{y_1} (-1)^{x_1 y_1} |y_1\rangle) \ldots (\Sigma_{y_n} (-1)^{x_n y_n} |y_n\rangle)$

$= \Sigma_y (-1)^{x_1 y_1 \oplus \ldots \oplus x_n y_n} |y_1 \ldots y_n\rangle$ ∎

42

- Introduction to quantum algorithms
- Parity problem and Deutsch's algorithm
- Constant vs. balanced problem
- Computing H⊗H⊗ ... ⊗H
- **Simon's problem**

43

Quantum vs. classical separations

Black-box problem	Quantum	Classical
Deutsch's problem	**1** (query)	**2** (queries)
constant vs. balanced	**1**	$\frac{1}{2}\,2^n + 1$
Simon's problem	$O(n)$	$\theta(2^{n/2})$

(only for exact)

(probabilistic)

44

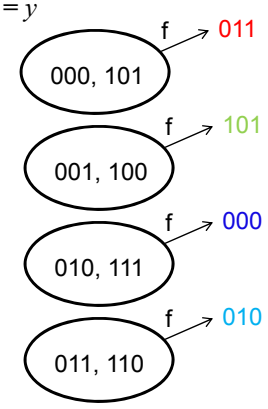
Simon's problem

Let $f: \{0,1\}^n \to \{0,1\}^n$ have the property that there exists an $r \in \{0,1\}^n$ such that $f(x) = f(y)$ iff $x \oplus y = r$ or $x = y$

Find ***r***.

Example:

x	$f(x)$
000	**011**
001	**101**
010	**000**
011	**010**
100	**101**
101	**011**
110	**010**
111	**000**



What is r is this case? r = **101**

A classical algorithm for Simon

Search for a ***collision***, an $x \neq y$ such that $f(x) = f(y)$

1. Choose $x_1, x_2, ..., x_k \in \{0,1\}^n$ uniformly randomly (independently)

2. For all $i \neq j$, if $f(x_i) = f(x_j)$ then output $x_i \oplus x_j$ and halt

A hard case is where r is chosen randomly from $\{0,1\}^n - \{0^n\}$ and then the "table" for f is filled out randomly subject to the structure implied by r

Question: How big does k have to be for the probability of a collision to be a constant, such as ¾?

Answer: order $2^{n/2}$ (each (x_i, x_j) collides with prob. $O(2^{-n})$)

Classical lower bound

Theorem: ***any*** classical algorithm solving Simon's problem must make $\Omega(2^{n/2})$ queries

Proof is omitted here

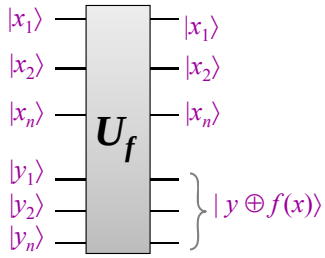
Note: the performance analysis of the previous algorithm does ***not*** imply the theorem

… how can we know that there isn't a ***different*** algorithm that performs better?

47

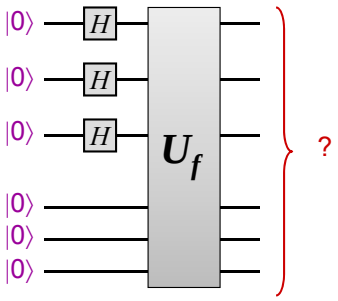
A quantum algorithm for Simon (1)

Queries:



Proposed start of quantum algorithm: query all values of f in superposition

What is the output state of this circuit?



48

A quantum algorithm for Simon (2)

Answer: the output state is $\sum_{x\in\{0,1\}^n} |x\rangle|f(x)\rangle$

Let $T \subseteq \{0,1\}^n$ be such that ***one*** element from each matched pair is in T (assume $r \neq 00...0$)

Example: could take $T = \{000, 001, 011, 111\}$

Then the output state can be written as:

$$\sum_{x\in T} |x\rangle|f(x)\rangle + |x\oplus r\rangle|f(x\oplus r)\rangle$$

$$= \sum_{x\in T} \left(|x\rangle + |x\oplus r\rangle\right)|f(x)\rangle$$

x	$f(x)$
000	011
001	101
010	000
011	010
100	101
101	011
110	010
111	000

49

A quantum algorithm for Simon (3)

Measuring the second register yields $|x\rangle + |x \oplus r\rangle$ in the first register, for a random $x \in T$

How can we use this to obtain ***some*** information about r ?

Try applying $H^{\otimes n}$ to the state, yielding:

$$\sum_{y\in\{0,1\}^n} (-1)^{x\bullet y}|y\rangle + \sum_{y\in\{0,1\}^n} (-1)^{(x\oplus r)\bullet y}|y\rangle$$

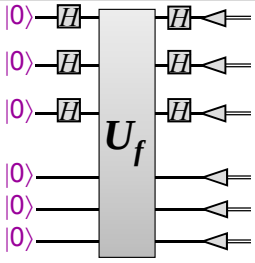
$$= \sum_{y\in\{0,1\}^n} (-1)^{x\bullet y}\left(1+(-1)^{r\bullet y}\right)|y\rangle$$

Measuring this state yields y with prob. $\begin{cases} (1/2)^{n-1} & \text{if } r\cdot y = 0 \\ 0 & \text{if } r\cdot y = 1 \end{cases}$

50

A quantum algorithm for Simon (4)

Executing this algorithm $k = n+O(1)$ times yields random $y_1, y_2, ..., y_k \in \{0,1\}^n$ such that $r \cdot y_1 = r \cdot y_2 = ... = r \cdot y_k = 0$



How does this help?

This is a system of k linear equations:

$$\begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1n} \\ y_{21} & y_{22} & \cdots & y_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ y_{k1} & y_{k2} & \cdots & y_{kn} \end{bmatrix} \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

With high probability, there is a unique non-zero solution that is r (which can be efficiently found by linear algebra)

51

Conclusion of Simon's algorithm

- Any classical algorithm has to query the black box $\Omega(2^{n/2})$ times, even to succeed with probability ¾.
- There is a quantum algorithm that queries the black box only $n+O(1)$ times, performs only $O(n^3)$ auxiliary operations (for the Hadamards, measurements, and linear algebra), and succeeds with probability ¾.

52

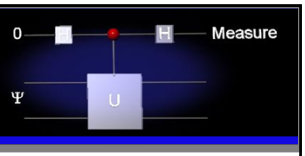
Aside:

- Note that we assumed that if we know how to classically compute f(x) then we can implement

$$|x\rangle|0\rangle \mapsto |x\rangle|f(x)\rangle$$

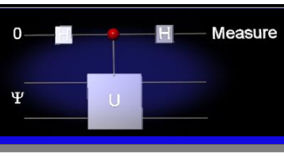
- Is that necessarily possible to do with comparable efficiency?

Irreversible gates from reversible ones

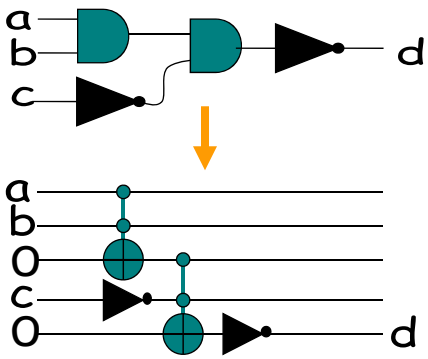


- Note that irreversible gates are really just reversible gates where we hardwire some inputs and throw away some outputs

Making reversible circuits

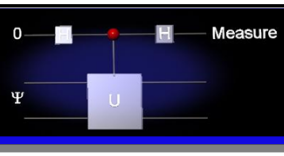


- Replace irreversible gates with their reversible counterparts



55

Making reversible circuits



- One problem is that there will be junk left in the extra bits
- Bennett showed how to “uncompute” the junk

$$|x\rangle|0\rangle|0\rangle|0\rangle$$

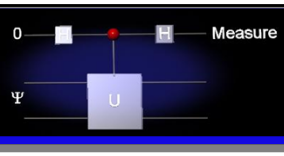
$$\xrightarrow{compute\ f(x)} |x\rangle|f(x)\rangle|junk(x)\rangle|0\rangle$$

$$\xrightarrow{copy\ f(x)} |x\rangle|f(x)\rangle|junk(x)\rangle|f(x)\rangle$$

$$\xrightarrow{uncompute\ f(x)} |x\rangle|0\rangle|0\rangle|f(x)\rangle$$

56

Making reversible circuits

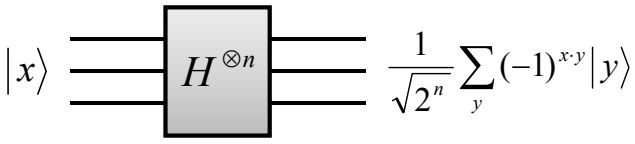


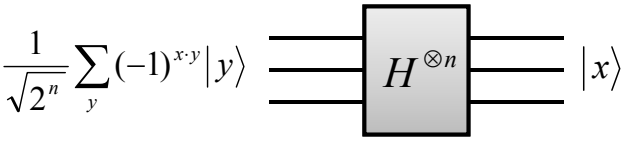
- An irreversible circuit with space S and depth (or "time") T can thus be simulated by a reversible circuit with space in O(S+T) and time O(T)
- Bennett also showed how to implement a reversible version with time $O(T^{1+\varepsilon})$ and space O(S log(T)) or time O(T) and space $O(ST^{\varepsilon})$.
- *Bottom line:* if we know how to classically compute f(x) then we can implement, with similar efficiency,

$$|x\rangle|0\rangle \mapsto |x\rangle|f(x)\rangle$$

57

Recall: Multi-qubit Hadamard





58

Quantum algorithms

- These algorithms have been computing essentially classical functions on quantum superpositions.
- When information is encoded in the phases of the basis states, measuring basis states would provide little useful information
- However, a quantum transformation might translate the phase information into information that is measurable in the computational basis.

59

Quantum phase estimation

- Suppose we wish to estimate a number $\omega \in [0,1)$ given the quantum state

$$\sum_{y=0}^{2^n-1} e^{2\pi i \omega y} |y\rangle$$

- Note that in binary we can express

$$\omega = 0.x_1 x_2 x_3 \ldots$$

$$2\omega = x_1 . x_2 x_3 \ldots$$

$$2^{n-1}\omega = x_1 x_2 x_3 \ldots x_{n-1} . x_n x_{n+1} \ldots$$

60

Quantum phase estimation

- Since $e^{2\pi ik} = 1$ for any integer *k*, we have

$$e^{2\pi i(2\omega)} = e^{2\pi i(x_1.x_2x_3\ldots)} = e^{2\pi i x_1} e^{2\pi i(0.x_2x_3\ldots)} = e^{2\pi i(0.x_2x_3\ldots)}$$

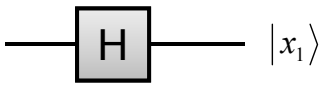
$$e^{2\pi i(2^k\omega)} = e^{2\pi i(0.x_{k+1}x_{k+2}\ldots)}$$

61

Quantum phase estimation

- If $\omega = 0.x_1$ then we can do the following

$$\frac{|0\rangle + e^{2\pi i(0.x_1)}|1\rangle}{\sqrt{2}} = \frac{|0\rangle + (-1)^{x_1}|1\rangle}{\sqrt{2}}$$



62

Useful identity

- We can show that

$$\sum_{y=0}^{2^n-1} e^{2\pi i \omega y} |y\rangle$$

$$= \left(|0\rangle + e^{2\pi i (2^{n-1}\omega)}|1\rangle\right) \otimes \left(|0\rangle + e^{2\pi i (2^{n-2}\omega)}|1\rangle\right) \otimes \cdots \otimes \left(|0\rangle + e^{2\pi i (\omega)}|1\rangle\right)$$

$$= \left(|0\rangle + e^{2\pi i (0.x_n x_{n+1} \ldots)}|1\rangle\right) \otimes \left(|0\rangle + e^{2\pi i (0.x_{n-1} x_n x_{n+1} \ldots)}|1\rangle\right) \otimes$$
$$\cdots \otimes \left(|0\rangle + e^{2\pi i (0.x_1 x_2 \ldots)}|1\rangle\right)$$

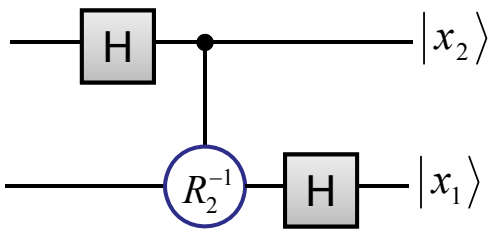
63

Quantum phase estimation

- So if $\omega = 0.x_1x_2$ then we can do the following

$$\frac{|0\rangle + e^{2\pi i (0.x_2)}|1\rangle}{\sqrt{2}}$$

$$\frac{|0\rangle + e^{2\pi i (0.x_1 x_2)}|1\rangle}{\sqrt{2}}$$

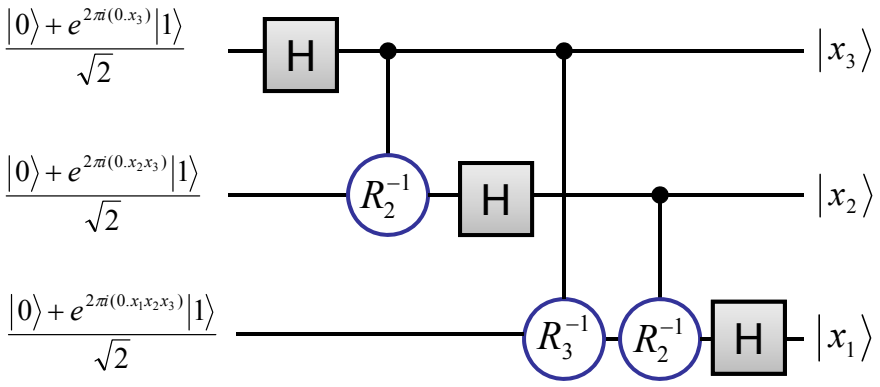


$$R_k = \begin{bmatrix} 1 & 0 \\ 0 & e^{2\pi i / 2^k} \end{bmatrix}$$

64

Quantum phase estimation

- So if $\omega = 0.x_1x_2x_3$ then we can do the following



65

Quantum phase estimation

- Generalizing this network (and reversing the order of the qubits at the end) gives us a network with $O(n^2)$ gates that implements

$$\sum_{y=0}^{2^n-1} e^{2\pi i \frac{x}{2^n} y} |y\rangle \mapsto |x\rangle$$

66

Discrete Fourier transform

- The discrete Fourier transform maps vectors of dimension *N* by transforming the **x**[th] elementary vector according to

$$(0,0,...,0,1,0,...0) \mapsto \frac{1}{\sqrt{N}}(1, e^{2\pi i \frac{x}{N}}, e^{2\pi i \frac{2x}{N}}, \ldots, e^{2\pi i \frac{(N-1)x}{N}})$$

- The quantum Fourier transform maps vectors in a Hilbert space of dimension *N* according to

$$|x\rangle \mapsto \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{2\pi i \frac{x}{N} y} |y\rangle$$

67

Discrete Fourier transform

- Thus we have illustrated how to implement (the inverse of) the quantum Fourier transform in a Hilbert space of dimension 2^n

68

Estimating arbitrary ω ϵ [0,1)

- What if ω is not necessarily of the form $\frac{x}{2^n}$ for some integer x ?
- The QFT will map

$$\sum_{x=0}^{2^n-1} e^{2\pi i \omega z} |z\rangle$$

to a superposition

$$|\tilde{\omega}\rangle = \sum_y \alpha_y |y\rangle$$

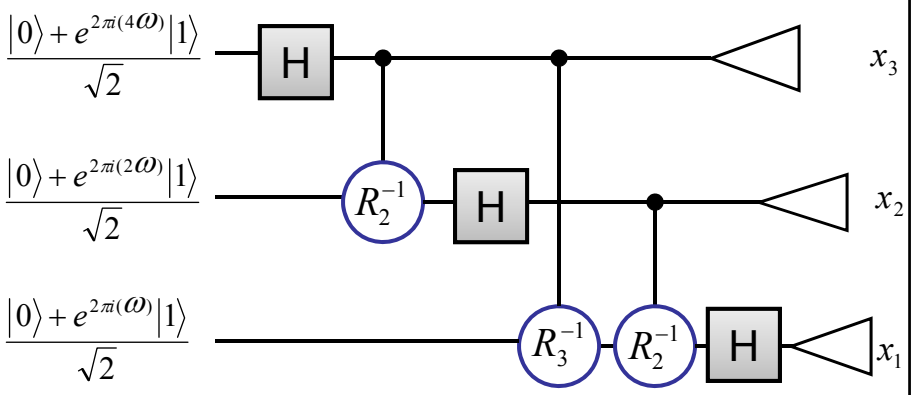
where

$$\Pr ob\left(\left|\frac{y}{N} - \omega\right| \le \frac{1}{N}\right) \ge \frac{8}{\pi^2} \qquad |\alpha_y| \in O\left(\frac{1}{\left|\frac{y}{N} - \omega\right|}\right)$$

69

Quantum phase estimation

- For any real $\omega \in [0,1)$

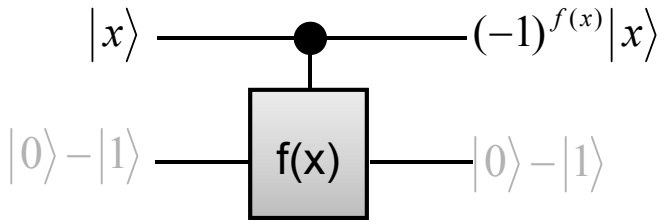


- With high probability $\frac{4x_1 + 2x_2 + x_3}{8} \approx \omega$

70

Eigenvalue kick-back

- Recall the "trick":

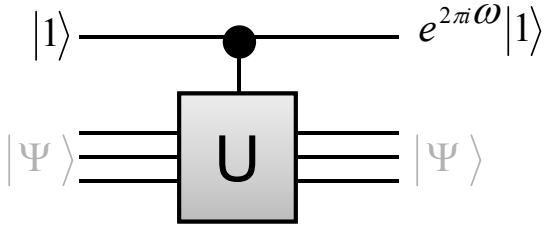


$$|x\rangle(|0\rangle - |1\rangle) \rightarrow |x\rangle(|f(x)\rangle - |f(x) \oplus 1\rangle)$$
$$= |x\rangle(-1)^{f(x)}(|0\rangle - |1\rangle)$$
$$= (-1)^{f(x)}|x\rangle(|0\rangle - |1\rangle)$$

71

Eigenvalue kick-back (Kitaev)

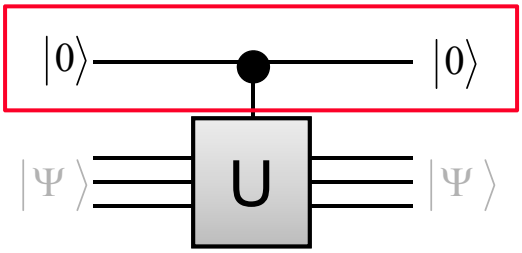
- Consider a unitary operation **U** with eigenvalue $e^{2\pi i\omega}$ and eigenvector $|\Psi\rangle$



$$|1\rangle|\Psi\rangle \rightarrow |1\rangle U|\Psi\rangle = |1\rangle e^{2\pi i\omega}|\Psi\rangle$$
$$= e^{2\pi i\omega}|1\rangle|\Psi\rangle$$

72

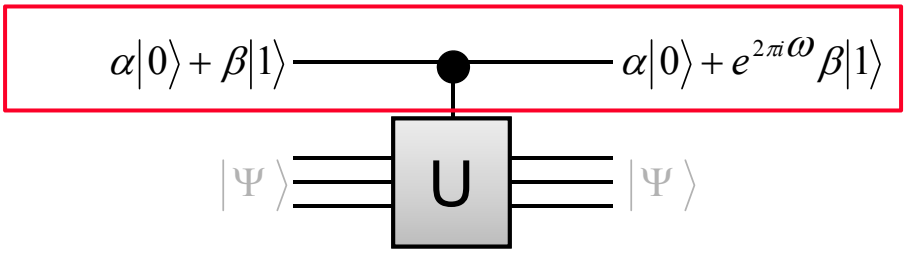
Eigenvalue kick-back



73

Eigenvalue kick-back

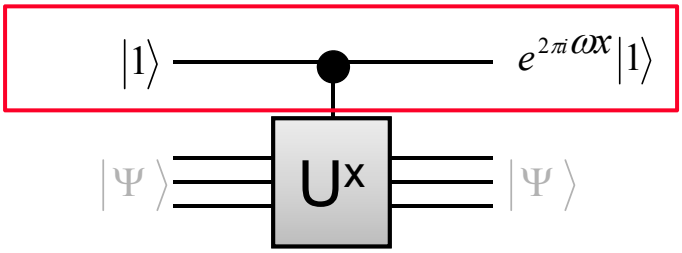
- As a relative phase, $e^{2\pi i\omega}$ becomes measurable



74

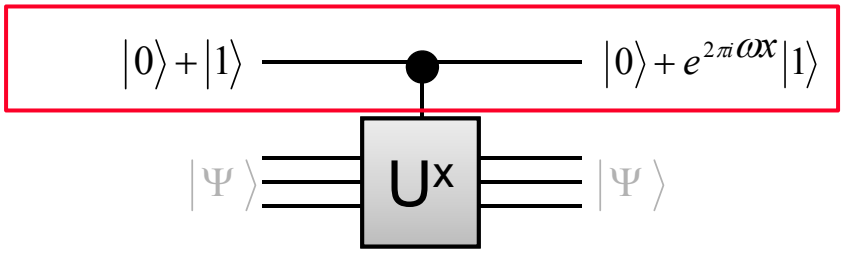
Eigenvalue kick-back

- If we exponentiate **U**, we get multiples of ω



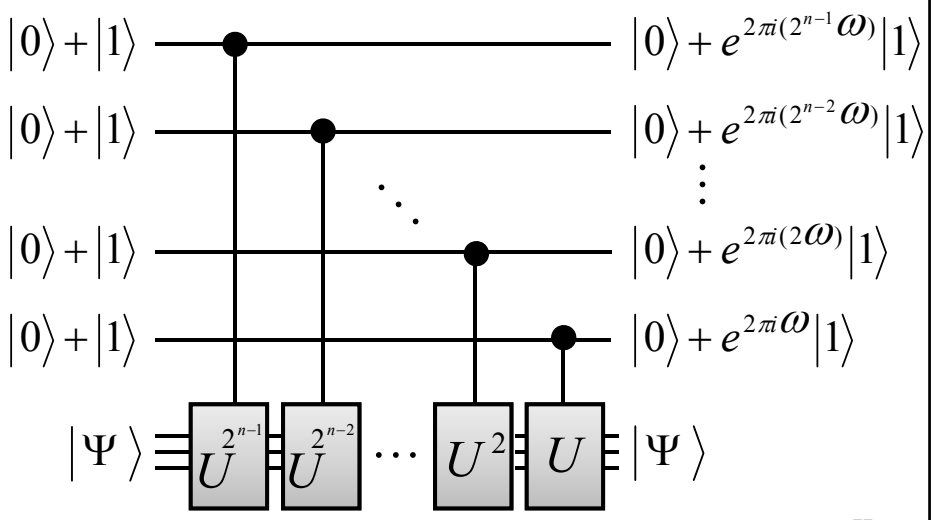
75

Eigenvalue kick-back



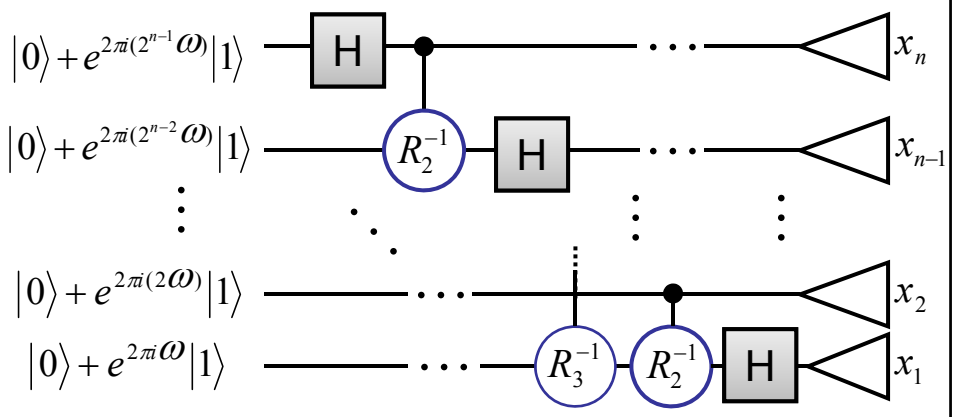
76

Eigenvalue kick-back



77

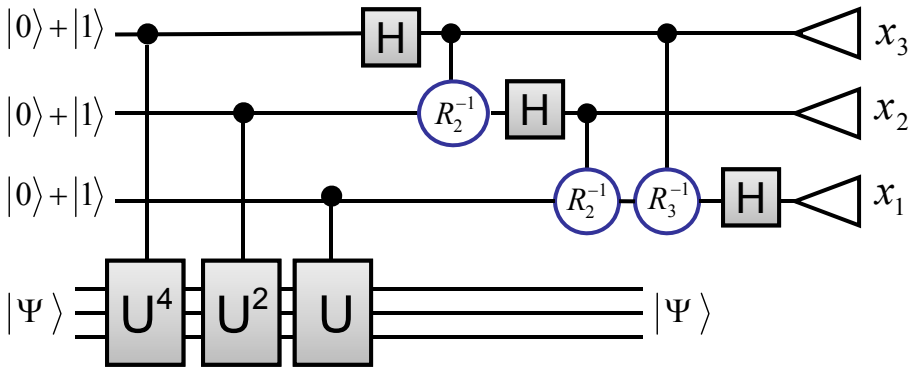
Phase estimation



$$\frac{2^{n-1}x_1 + 2^{n-2}x_2 + \cdots x_n}{2^n} \approx \omega$$

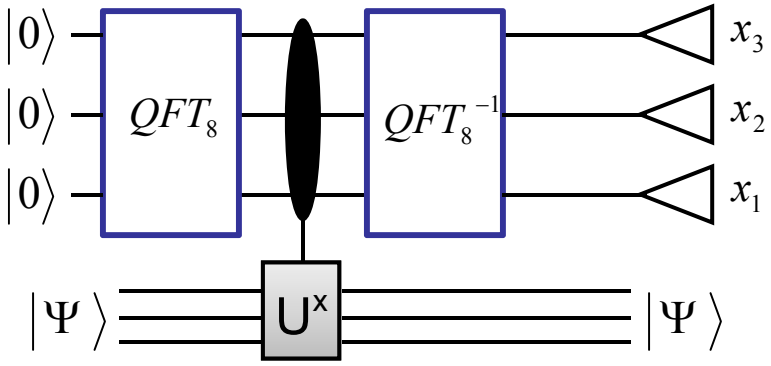
78

Eigenvalue estimation



79

Eigenvalue estimation



80

Eigenvalue estimation

- Given **U** with eigenvector $|\Psi\rangle$ and eigenvalue $e^{2\pi i\omega}$, we thus have an algorithm that maps

$$|0\rangle|\Psi\rangle \rightarrow |\tilde{\omega}\rangle|\Psi\rangle$$

81

Eigenvalue estimation

- Given **U** with eigenvectors $|\Psi_k\rangle$ and respective eigenvalues $e^{2\pi i\omega_k}$ we thus have an algorithm that maps

$$|0\rangle|\Psi_k\rangle \mapsto |\tilde{\omega}_k\rangle|\Psi_k\rangle$$

and therefore

$$|0\rangle\sum_k \alpha_k|\Psi_k\rangle = \sum_k \alpha_k|0\rangle|\Psi_k\rangle \mapsto \sum_k \alpha_k|\tilde{\omega}_k\rangle|\Psi_k\rangle$$

82

Eigenvalue estimation

- Measuring the first register of

$$\sum_k \alpha_k \left| \tilde{\omega}_k \right\rangle \left| \Psi_k \right\rangle$$

is equivalent to measuring $\left| \tilde{\omega}_k \right\rangle$ with probability $\left| \alpha_k \right|^2$.

83

Example

- Suppose we have a group $\mathbf{G}$ and we wish to find the order of $a \in \mathbf{G}$ (i.e. the smallest positive r such that $a^r \equiv 1$).
- If we can efficiently do arithmetic in the group, then we can realize a unitary operator $\mathbf{U_a}$ that maps $\left| x \right\rangle \mapsto \left| ax \right\rangle$.
- Notice that

$$U_a{}^r = U_{a^r} = I$$

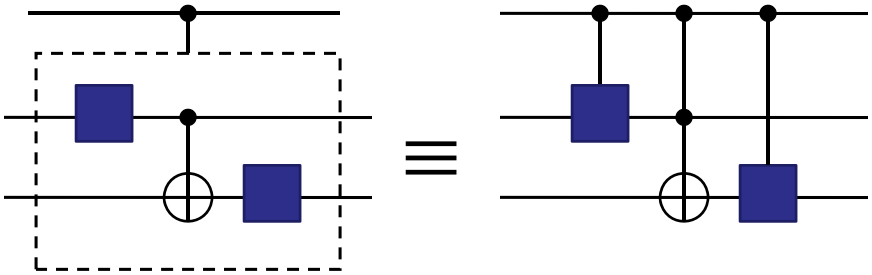
- This means that the eigenvalues of $\mathbf{U_a}$ are of the form $e^{2\pi i \frac{k}{r}}$ where k is an integer.

84

How do we implement c-U ?

Replace every gate G in the circuit with a c-G.

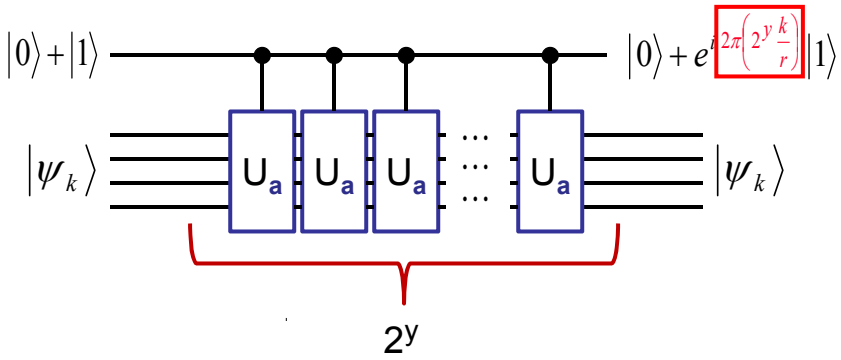
For example,



85

Inefficient exponentiation

We can effect a relative phase shift of $e^{i\left[2\pi\left(2^y \frac{k}{r}\right)\right]}$

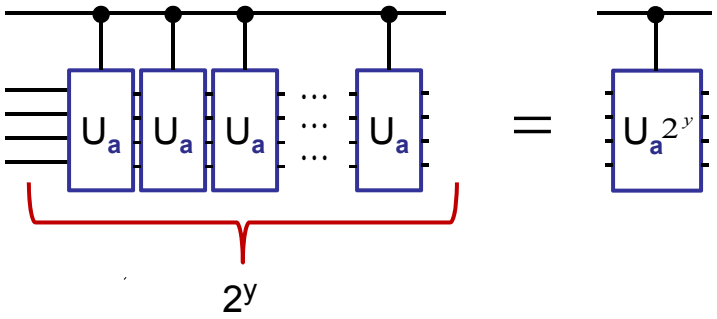


86

Efficient exponentiation

But we can also do it **efficiently** by noticing that

$$U_a^{2^y} = U_{a^{2^y}}$$



87

Quantum factoring

- The security of many public key cryptosystems used in industry today relies on the difficulty of factoring large numbers into smaller factors.
- Factoring the integer *N* into smaller factors can be reduced to the following task:

Given integer a, find the smallest positive integer r so that $a^r \equiv 1 \bmod N$

88

(Aside: how does factoring reduce to order-finding ?)

- The most common approach for factoring integers is the difference of squares technique:
  - "Randomly" find two integers *x* and *y* satisfying

$$x^2 = y^2 \bmod N$$

  - So *N* divides $x^2 - y^2 = (x-y)(x+y)$
  - Hope that $\gcd(N, x-y)$ is non-trivial

- If *r* is even, then let

$$x = a^{r/2} \bmod N$$

so that

$$x^2 = 1^2 \bmod N$$

89

Quantum factoring

Since we know how to efficiently multiply by ***a*** mod *N*, we can efficiently implement

$$U_a|x\rangle = |ax\rangle$$

Note that

$$U_{a^r}|x\rangle = |a^r x\rangle = |x\rangle$$

i.e.

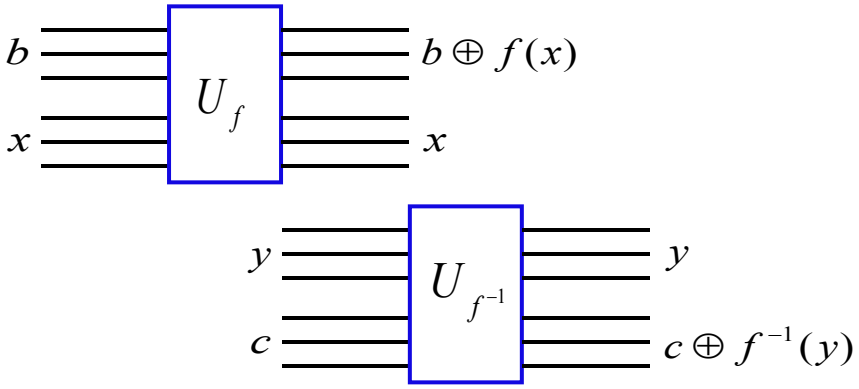
$$U_a^r = \mathrm{I}$$

Remember that $|x\rangle$ represents the state corresponding to the binary representation of x (e.g. for four qubits, $|2\rangle$ represents $|0010\rangle$)

90

(Aside : more on reversible computing)

If we know how to efficiently compute f and f^{-1} then we can efficiently and reversibly map

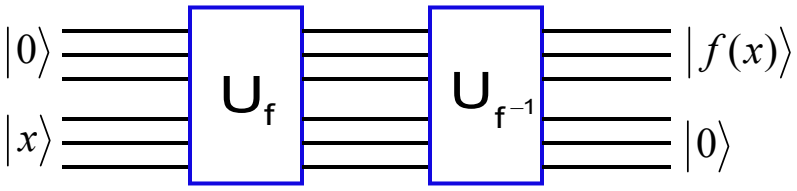


91

(Aside : more on reversible computing)

And therefore we can efficiently map

$$|x\rangle \mapsto |f(x)\rangle$$



92

Interesting eigenvalues

If $U_a^r = I$ then the eigenvalues of U_a are of the form

$$e^{2\pi i \frac{k}{r}}$$

$$\Rightarrow \quad U_a \left|\psi_k\right\rangle = \mathrm{e}^{\mathrm{i}2\pi\frac{\mathrm{k}}{r}} \left|\psi_k\right\rangle$$

$$\left|\psi_k\right\rangle = \sum_{j=0}^{r-1} e^{i2\pi j\frac{k}{r}} \left|a^j\right\rangle$$

93

Checking the eigenvalues

$$U_a \left|\psi_k\right\rangle = \sum_{j=0}^{r-1} e^{-i2\pi j\frac{k}{r}} U_a \left|a^j\right\rangle$$

$$= \sum_{j=0}^{r-1} e^{-i2\pi j\frac{k}{r}} \left|a^{j+1}\right\rangle = e^{i2\pi\frac{k}{r}} \left(\sum_{j=1}^{r} e^{-i2\pi j\frac{k}{r}} \left|a^j\right\rangle \right)$$

$$= \mathrm{e}^{\mathrm{i}2\pi\frac{\mathrm{k}}{\mathrm{r}}} \left(\sum_{\mathrm{j}=0}^{\mathrm{r}-1} \mathrm{e}^{-\mathrm{i}2\pi j\frac{\mathrm{k}}{\mathrm{r}}} \left|\mathrm{a}^{\mathrm{j}}\right\rangle \right) = e^{i2\pi\frac{k}{r}} \left|\psi_k\right\rangle$$

94

Finding r

For most integers k, a good estimate of $\frac{k}{r}$ (with error at most $\frac{1}{2r^2}$) allows us to determine r (even if we don't know k).

(using continued fractions)

95

Complexity comparison

•The best known rigorous classical algorithms use

$$e^{O(\sqrt{\log(N)\log\log(N)})}$$

operations

•The best known heuristic classical algorithms use

$$e^{O((\log(N)^{\frac{1}{3}}\log\log(N)^{\frac{2}{3}})}$$

operations

•The quantum algorithm uses poly(log(N))

$$= e^{O(\log\log(N))}$$

operations

96

Shor's Factoring Algorithm

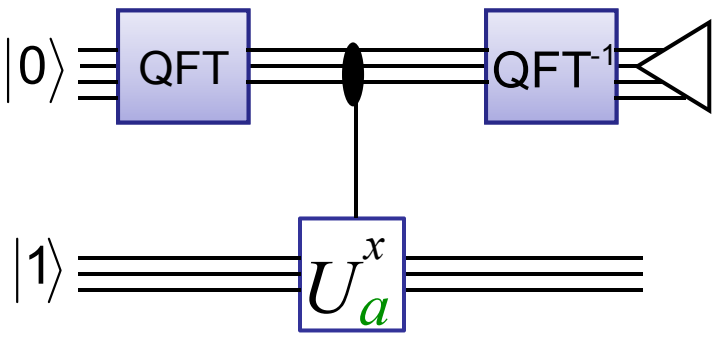
$$\sum_x |x\rangle|1\rangle \mapsto \sum_x |x\rangle|a^x\rangle$$

$$= \sum_{w=0}^{r-1} \sum_y |w + yr\rangle|a^w\rangle$$

$$\xrightarrow{\text{QFT}^{-1}} \sum_w \left(\bigwedge_{\frac{0}{r}} \bigwedge_{\frac{1}{r}} \cdots \bigwedge_{\frac{k}{r}} \cdots \right) |a^w\rangle$$

97

A circuit for Shor's Factoring Algorithm



98

Eigenvalue Estimation Factoring Algorithm

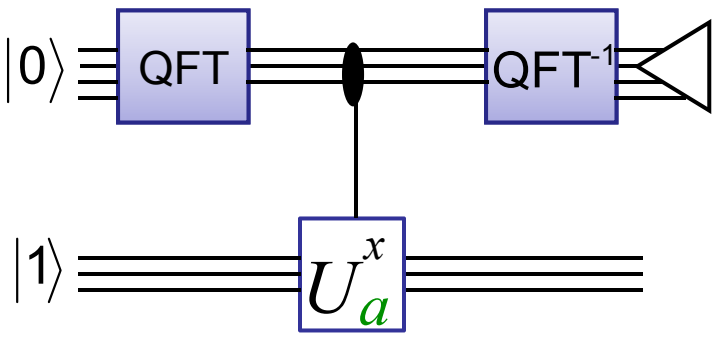
$$|0\rangle|1\rangle = \sum_{k=0}^{r-1} |0\rangle|\psi_k\rangle \mapsto \sum_{k=0}^{r-1}\sum_{x} |x\rangle|\psi_k\rangle$$

$$\mapsto \sum_{k=0}^{r-1}\sum_{x} e^{2\pi i k x/r} |x\rangle|\psi_k\rangle$$

$$\Rightarrow \sum_{k} \left(\int_{\frac{k}{r}} \right) |\psi_k\rangle$$

99

A circuit for Eigenvalue Estimation Factoring Algorithm



100

Equivalence

$$\sum_x |x\rangle|1\rangle = \sum_{k=0}^{r-1} \sum_x |x\rangle|\psi_k\rangle$$

$$\sum_{w=0}^{r=1} \sum_y |w+yr\rangle|a^w\rangle = \sum_{k=0}^{r-1} \sum_x e^{2\pi i k x/r} |x\rangle|\psi_k\rangle$$

$$\sum_w \left(\bigwedge_{\frac{0}{r}} \bigwedge_{\frac{1}{r}} \bigwedge_{\frac{k}{r}} \bigwedge \right) |a^w\rangle = \sum_k \left(\bigwedge_{\frac{k}{r}} \right) |\psi_k\rangle$$

101

Discrete Logarithm Problem

Consider two elements $a, b \in G$ from a group G satisfying

$$a^r = 1$$
$$b = a^s$$

Find s.

$$U_a |x\rangle = |ax\rangle$$

102

Discrete Logarithm Problem

We know U_a has eigenvectors

$$\left|\psi_k\right\rangle = \sum_{j=0}^{r-1} e^{-i2\pi j\frac{k}{r}} \left|a^j\right\rangle$$

$$U_a\left|\psi_k\right\rangle = e^{i2\pi\frac{k}{r}}\left|\psi_k\right\rangle$$

103

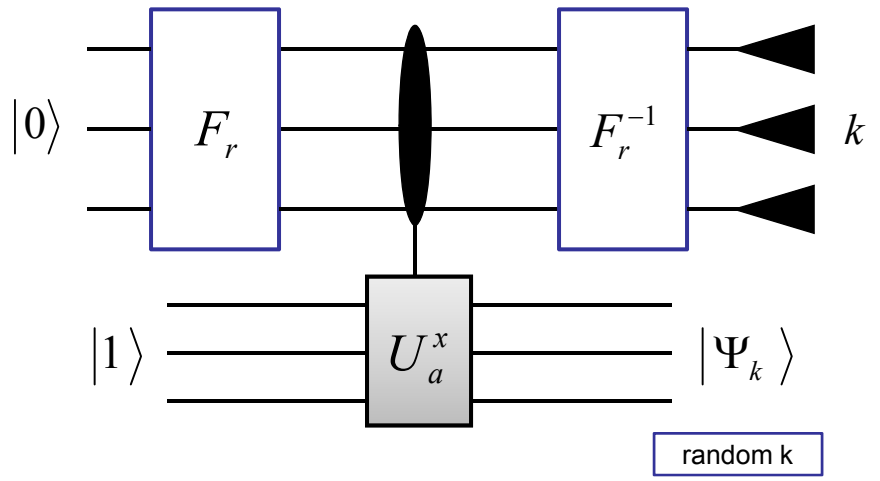
Discrete Logarithm Problem

Thus U_b has the same eigenvectors but with eigenvalues exponentiated to the power of s

$$U_b\left|\psi_k\right\rangle = U_a{}^s\left|\psi_k\right\rangle = e^{i2\pi\frac{ks}{r}}\left|\psi_k\right\rangle$$

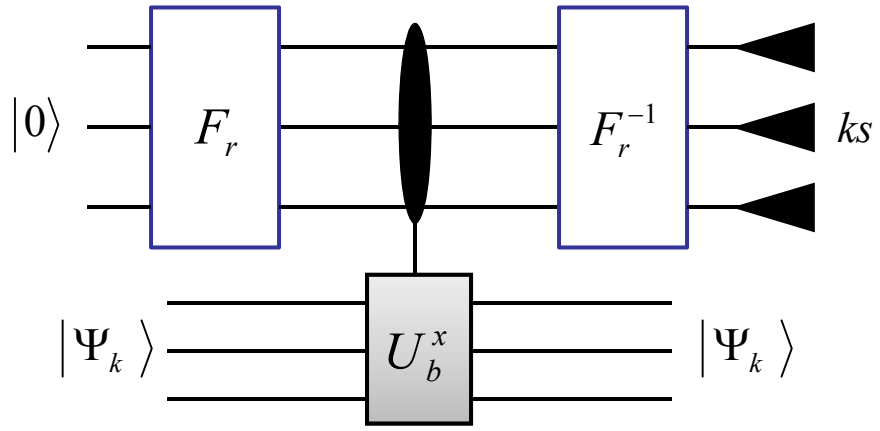
104

Discrete Logarithm Problem



105

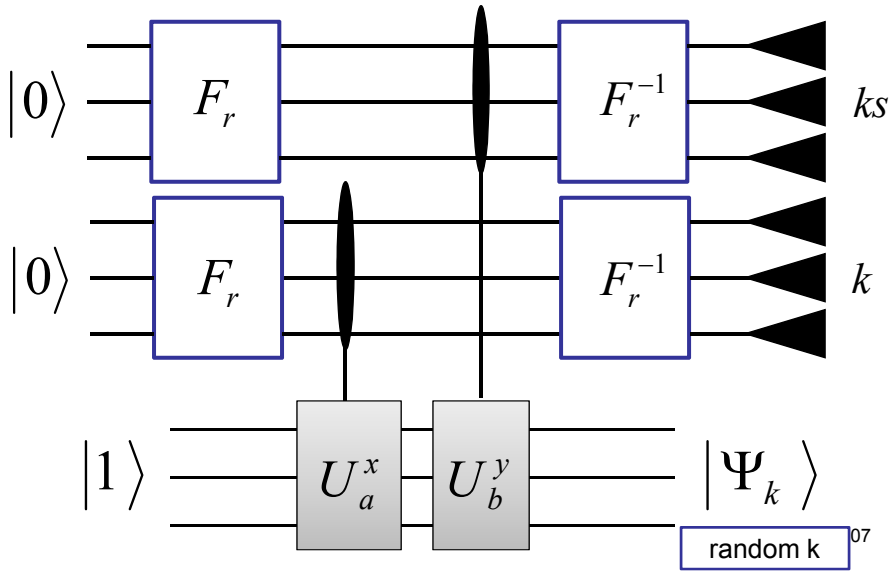
Discrete Logarithm Problem



Given k and ks, we can compute $s \bmod r$
(provided k and r are coprime)

106

Complete Circuit



07

Generalization of Simon's problem, order-finding and DLP: "Hidden subgroup problem"

- A unifying framework was developed for these problems

$$f : G \rightarrow X$$

$$f(x) = f(y) \quad \text{iff} \quad x + S = y + S$$

for some $S \leq G$

- If G is Abelian, finitely generated, and represented in a reasonable way, we can efficiently find S.

108

Example (I)

Deutsch's Problem:

$$G = \{0,1\} \qquad X = \{0,1\}$$

$$S = \{0\} \text{ or } \{0,1\}$$

Order finding:

$$G = Z \qquad X \text{ any group}$$

$$f(x) = a^x$$

$$S = rZ$$

109

Example (II)

Discrete Log of $b=a^k$ to base a :

$$G = Z_r \times Z_r \qquad X \text{ any group}$$

$$f(x,y) = a^x b^y$$

$$S = \langle (k,-1) \rangle$$

110

What about non-Abelian HSP

- Consider the symmetric group $G = S_n$
- S_n is the set of permutations of n elements
- Let G be an n-vertex graph
- Let $X_G = \{\pi(G) \mid \pi \in S_n\}$
- Define $f_G : S_n \to X_G \qquad f_G(\pi) = \pi(G)$
- Then $f_G(\pi_1) = f_G(\pi_2) \Leftrightarrow \pi_1 S = \pi_2 S$

  where $S = AUT(G) = \{\pi \mid \pi(G) = G\}$

Further reading

http://math.nist.gov/quantum/zoo/ *(maintained by S. Jordan)*


Quantum algorithms for algebraic problems

Andrew M. Childs*

Department of Combinatorics & Optimization and Institute for Quantum Computing
University of Waterloo, Waterloo, Ontario, Canada N2L 3G1

Wim van Dam†

Departments of Computer Science and Physics
University of California, Santa Barbara, California 93106, USA

Contents


Quantum Algorithms

Michele Mosca
Institute for Quantum Computing and Dept. of Combinatorics & Optimization
University of Waterloo and St. Jerome's University,
and Perimeter Institute for Theoretical Physics
www.iqc.ca/~ mmosca/web

Article Outline

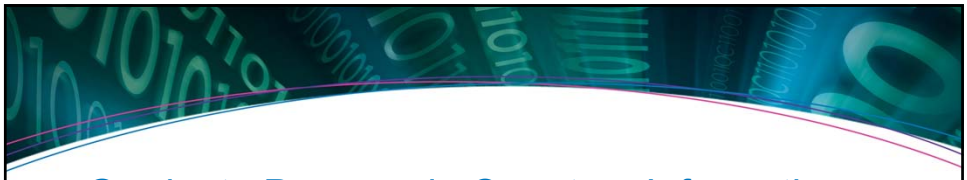
Glossary

1. Definition of the Subject and Its Importance
2. Introduction and Overview
3. The Early Quantum Algorithms
4. Factoring, Discrete Logarithms, and the Abelian Hidden Subgroup Problem
5. Algorithms based on Amplitude Amplification
6. Simulation of Quantum Mechanical Systems
7. Generalizations of the Abelian Hidden Subgroup Problem
8. Quantum Walk Algorithms
9. Adiabatic Algorithms
10. Topological Algorithms
11. Quantum algorithms for quantum tasks
12. Future Directions


Algorithms for Quantum Computers

Jamie Smith and Michele Mosca

1 Introduction



Graduate Program in Quantum Information

- A cutting-edge interdisciplinary program leading to MMath, MSc, MASc and PhD degrees (e.g. PhD in Applied Mathematics (Quantum Information))
- Students must apply directly to one of the following academic units:
  - Applied Mathematics
  - Chemistry
  - Combinatorics and Optimization
  - David R. Cheriton School of Computer Science
  - Electrical and Computer Engineering
  - Physics and Astronomy
- Applicants will be subject to the normal admission procedures of the home unit
- Students will be subject to the normal program requirements of the home unit, plus additional QI requirements
- Several scholarships available to both domestic and international students
- For more information, visit www.iqc.ca or email grad@iqc.ca

UNIVERSITY OF WATERLOO | IQC Institute for Quantum Computing