
Lecture 13 – Quantum computers, Shor's algorithm, post-quantum cryptography

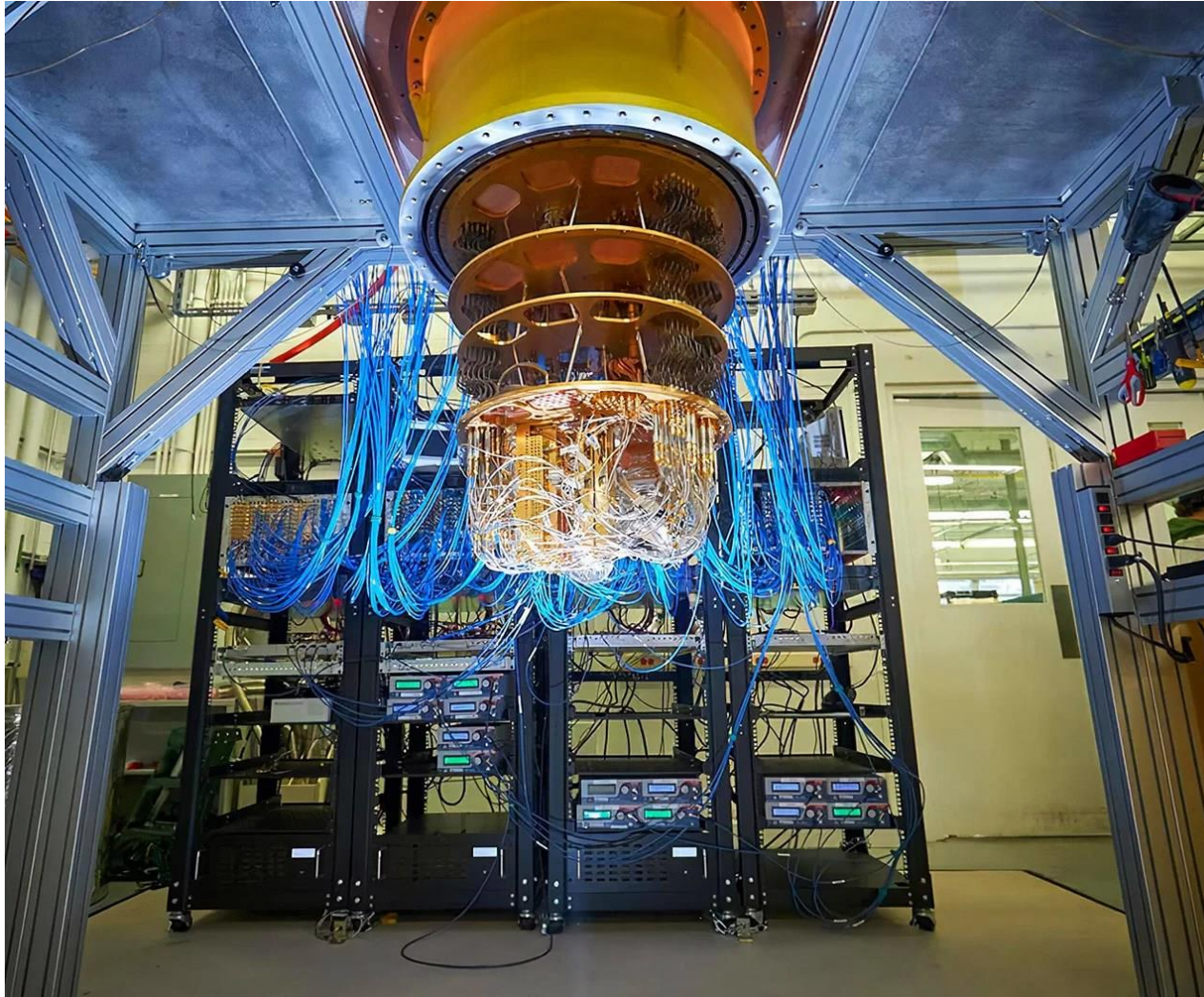
TEK4500

27.11.2024

Håkon Jacobsen

hakon.jacobsen@its.uio.no

Quantum computers



Quantum computers – the beginning

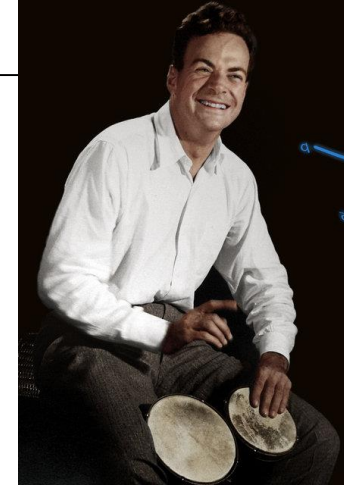
International Journal of Theoretical Physics, Vol. 21, Nos. 6/7, 1982

Simulating Physics with Computers

Richard P. Feynman

Department of Physics, California Institute of Technology, Pasadena, California 91107

Received May 7, 1981



Qubits

- Classical bit:



- Qubit:

superposition



$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle$$

amplitudes



$$\alpha, \beta \in \mathbb{C}$$

$$|\alpha|^2 + |\beta|^2 = 1$$

We cannot observe α and β directly!

- Can **measure** $|\psi\rangle \Rightarrow$ state collapses *randomly* to either $|0\rangle$ or $|1\rangle$ (with what probability?)

$$\Pr[|\psi\rangle \text{ collapses to } |0\rangle] = |\alpha|^2$$

$$\Pr[|\psi\rangle \text{ collapses to } |1\rangle] = |\beta|^2$$

$$|\psi\rangle = 0.6|0\rangle - 0.8|1\rangle$$

$$|0.6|^2 = 0.36$$

$$|-0.8|^2 = 0.64$$

Multiple qubits

$$|\psi\rangle = \alpha |00\rangle + \beta |01\rangle + \gamma |10\rangle + \delta |11\rangle$$

$$\alpha, \beta, \gamma, \delta \in \mathbb{C}$$

$$|\alpha|^2 + |\beta|^2 + |\gamma|^2 + |\delta|^2 = 1$$

- Can measure $|\psi\rangle \Rightarrow$ state collapses randomly to $|00\rangle$, $|01\rangle$, $|10\rangle$ or $|11\rangle$ (with probability $|\alpha|^2$, $|\beta|^2$, ...)
- Can measure *just* the first qubit in $|\psi\rangle$
 - $\Rightarrow$ state collapses randomly to $\alpha'|00\rangle + \beta'|01\rangle$ or to $\gamma'|10\rangle + \delta'|11\rangle$ (with probability $|\alpha|^2 + |\beta|^2$ or $|\gamma|^2 + |\delta|^2$)
- Can combine two independent qubits:

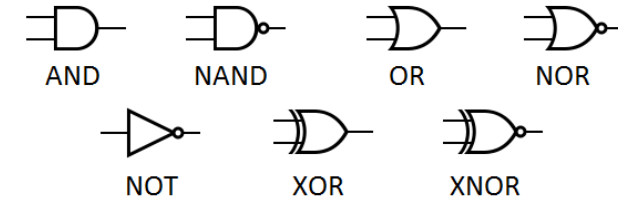
$$|\psi_0\rangle|\psi_1\rangle = (\alpha|0\rangle + \beta|1\rangle)(\gamma|0\rangle + \delta|1\rangle) = \alpha\gamma|00\rangle + \alpha\delta|01\rangle + \beta\gamma|10\rangle + \beta\delta|11\rangle$$

- $|\psi\rangle = \alpha_0|000\rangle + \alpha_1|001\rangle + \alpha_2|010\rangle + \alpha_3|011\rangle + \alpha_4|100\rangle + \alpha_5|101\rangle + \alpha_6|110\rangle + \alpha_7|111\rangle$
- N -qubit system: 2^N basis states

Quantum gates

- Can change classical bits with logical gates

$$0 \xrightarrow{\text{NOT}} 1$$



- Can change qubits with **quantum gates**

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \xrightarrow{U} |\psi'\rangle = \alpha'|0\rangle + \beta'|1\rangle$$

- Constraints on U :

- Linearity: $U(\alpha|0\rangle + \beta|1\rangle) = \alpha \cdot U(|0\rangle) + \beta \cdot U(|1\rangle)$
- $|\alpha'|^2 + |\beta'|^2 = 1 \implies U$ is invertible

Operator	Gate(s)
Pauli-X (X)	
Pauli-Z (Z)	
Hadamard (H)	
Controlled Not (CNOT, CX)	

(Quantum) NOT-gate

$$X(|0\rangle) = |1\rangle$$

$$X(|1\rangle) = |0\rangle$$

$$X(|\psi\rangle) = X(\alpha|0\rangle + \beta|1\rangle) = \alpha \cdot X(|0\rangle) + \beta \cdot X(|1\rangle) = \alpha|1\rangle + \beta|0\rangle$$

Hadamard gate

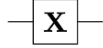
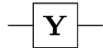
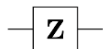
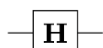
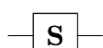
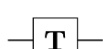
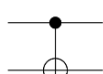
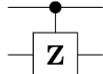
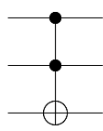
$$\mathbf{H}(|0\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \stackrel{\text{def}}{=} |+\rangle$$

$$\mathbf{H}(|1\rangle) = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \stackrel{\text{def}}{=} |-\rangle$$

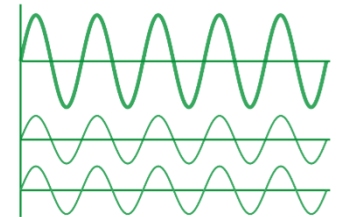
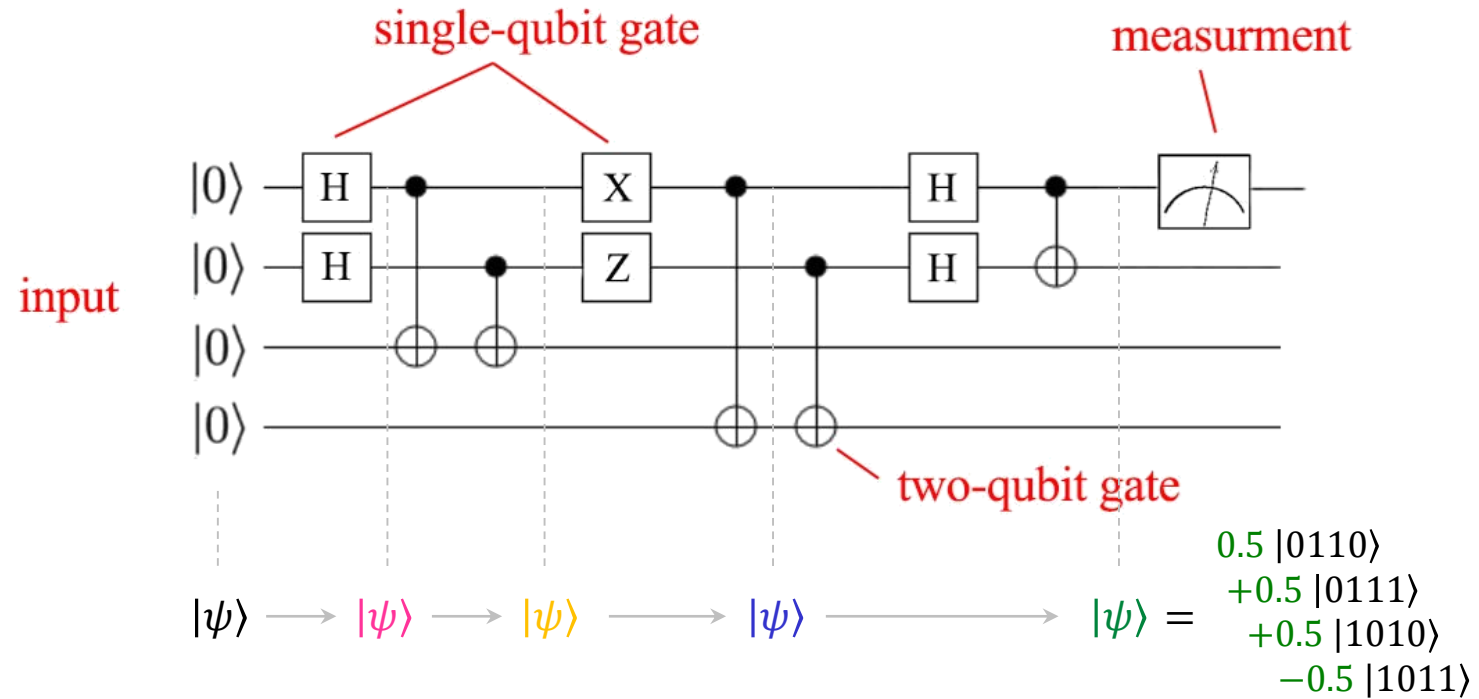
$$\mathbf{H}(|\psi\rangle) = \mathbf{H}(\alpha|0\rangle + \beta|1\rangle) = \alpha \cdot \mathbf{H}(|0\rangle) + \beta \cdot \mathbf{H}(|1\rangle) = \alpha|+\rangle + \beta|-\rangle$$

$$\Pr[|-\rangle \text{ collapse to } |1\rangle] = \left| \frac{-1}{\sqrt{2}} \right|^2 = 0.5$$

(Infinitely) many more gates...

Operator	Gate(s)	Matrix
Pauli-X (X)	 	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
Pauli-Y (Y)		$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$
Pauli-Z (Z)		$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
Hadamard (H)		$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
Phase (S, P)		$\begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$
$\pi/8$ (T)		$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}$
Controlled Not (CNOT, CX)		$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$
Controlled Z (CZ)	 	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$
SWAP	 	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
Toffoli (CCNOT, CCX, TOFF)		$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$

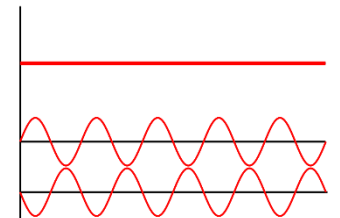
Quantum computers



What makes quantum computation special?

Interference

$+0 \cdot |0000\rangle$
 $+0 \cdot |0001\rangle$
 $+0 \cdot |0010\rangle$
 $\vdots$



The Deutsch–Jozsa's algorithm

$\mathbf{x}$	$f(\mathbf{x})$
00	1
01	1
10	1
11	1

constant

vs.

$\mathbf{x}$	$f(\mathbf{x})$
00	0
01	1
10	1
11	0

balanced

The Deutsch–Jozsa's algorithm

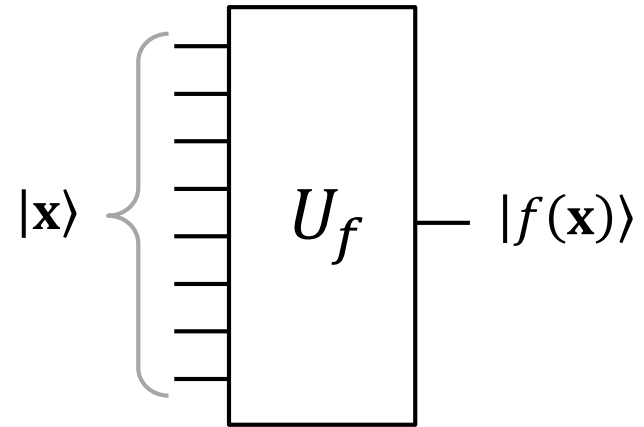
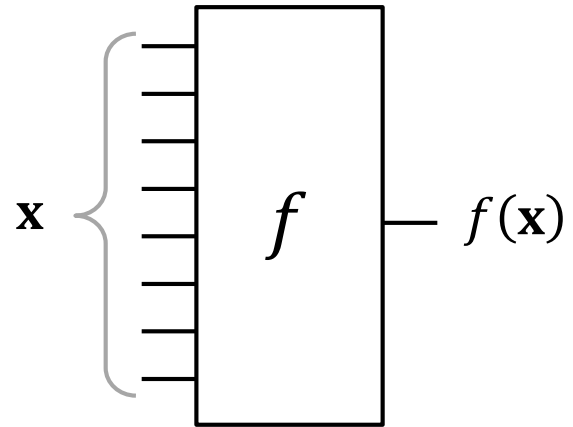
- Classic algorithm must test $2^{n/2} + 1$ inputs in the worst case

x	$f(x)$
00	0
01	0
10	1
11	

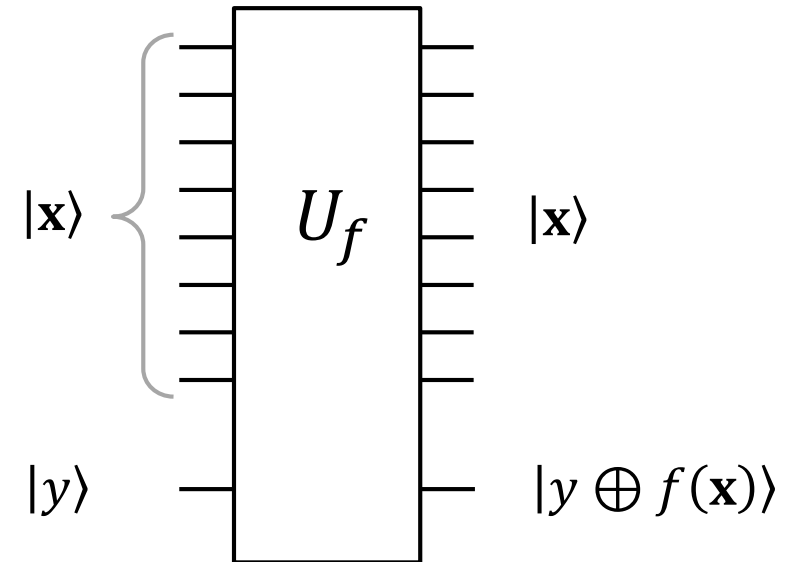
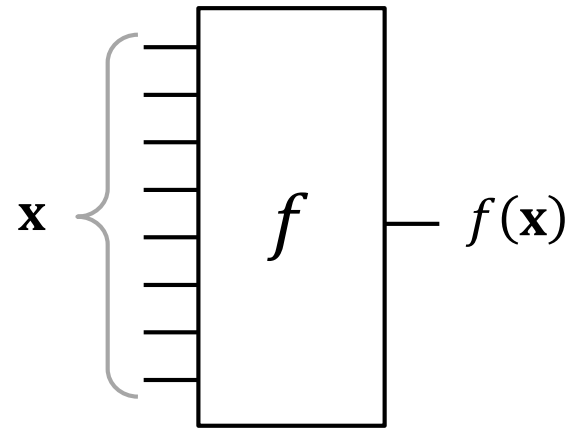
balanced

- Can a quantum computer do better?

The Deutsch–Jozsa's algorithm



The Deutsch–Jozsa's algorithm

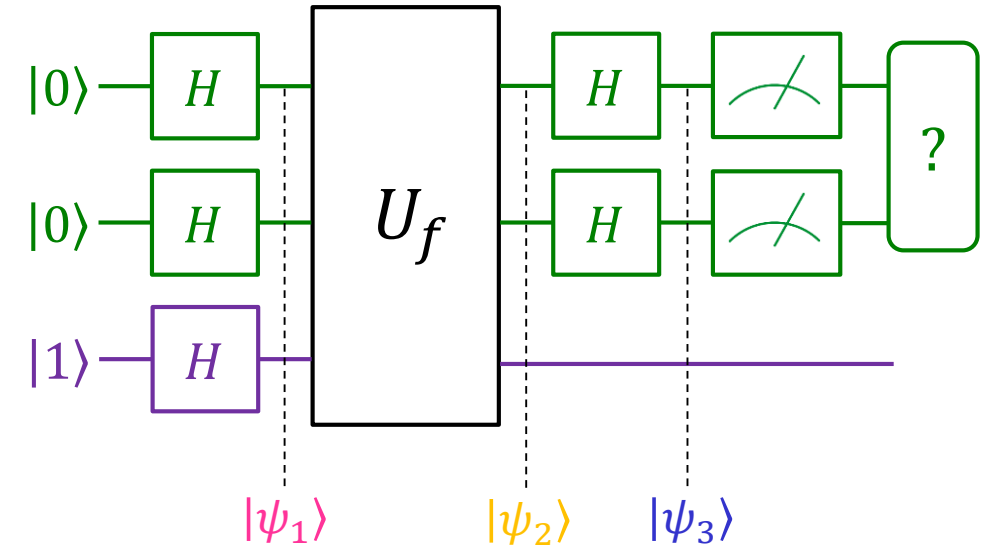


The Deutsch–Jozsa's algorithm

$$\begin{aligned}
 |\psi_1\rangle &= |+\rangle|+\rangle|-\rangle = (|0\rangle + |1\rangle)(|0\rangle + |1\rangle)(|0\rangle - |1\rangle) \\
 &= |00\rangle|0\rangle + |01\rangle|0\rangle + |10\rangle|0\rangle + |11\rangle|0\rangle \\
 &\quad - |00\rangle|1\rangle - |01\rangle|1\rangle - |10\rangle|1\rangle - |11\rangle|1\rangle
 \end{aligned}$$

$$\begin{aligned}
 |\psi_2\rangle &= |00\rangle|0 \oplus f(00)\rangle - |00\rangle|1 \oplus f(00)\rangle \quad (-1)^{f(00)}|00\rangle(|0\rangle - |1\rangle) \\
 &\quad + |01\rangle|0 \oplus f(01)\rangle - |01\rangle|1 \oplus f(01)\rangle \quad (-1)^{f(01)}|01\rangle(|0\rangle - |1\rangle) \\
 &\quad + |10\rangle|0 \oplus f(10)\rangle - |10\rangle|1 \oplus f(10)\rangle \quad \vdots \\
 &\quad + |11\rangle|0 \oplus f(11)\rangle - |11\rangle|1 \oplus f(11)\rangle \\
 &= \left((-1)^{f(00)}|00\rangle + (-1)^{f(01)}|01\rangle + (-1)^{f(10)}|10\rangle + (-1)^{f(11)}|11\rangle \right) (|0\rangle - |1\rangle)
 \end{aligned}$$

$$|\psi_3\rangle = \begin{pmatrix} (-1)^{f(00)}|+\rangle|+\rangle \\ + \\ (-1)^{f(01)}|+\rangle|-\rangle \\ + \\ (-1)^{f(10)}|-\rangle|+\rangle \\ + \\ (-1)^{f(11)}|-\rangle|-\rangle \end{pmatrix} (|0\rangle - |1\rangle) = \begin{pmatrix} (-1)^{f(00)}(|00\rangle + |01\rangle + |10\rangle + |11\rangle) \\ + \\ (-1)^{f(01)}(|00\rangle - |01\rangle + |10\rangle - |11\rangle) \\ + \\ (-1)^{f(10)}(|00\rangle + |01\rangle - |10\rangle - |11\rangle) \\ + \\ (-1)^{f(11)}(|00\rangle - |01\rangle - |10\rangle + |11\rangle) \end{pmatrix} (|0\rangle - |1\rangle)$$

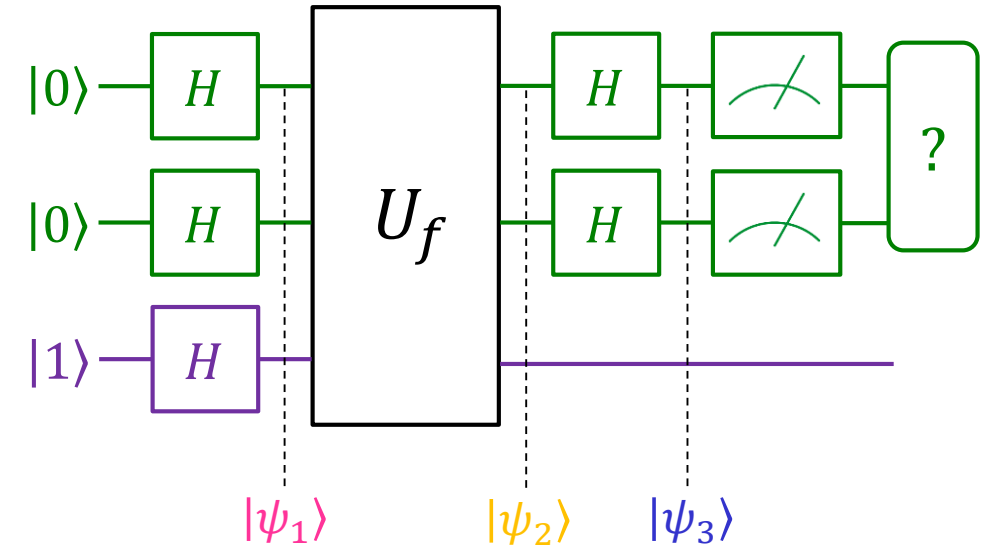


The Deutsch–Jozsa's algorithm

$$\begin{aligned}
 |\psi_1\rangle &= |+\rangle|+\rangle|-\rangle = (|0\rangle + |1\rangle)(|0\rangle + |1\rangle)(|0\rangle - |1\rangle) \\
 &= |00\rangle|0\rangle + |01\rangle|0\rangle + |10\rangle|0\rangle + |11\rangle|0\rangle \\
 &\quad - |00\rangle|1\rangle - |01\rangle|1\rangle - |10\rangle|1\rangle - |11\rangle|1\rangle
 \end{aligned}$$

$$\begin{aligned}
 |\psi_2\rangle &= |00\rangle|0 \oplus f(00)\rangle - |00\rangle|1 \oplus f(00)\rangle \\
 &\quad + |01\rangle|0 \oplus f(01)\rangle - |01\rangle|1 \oplus f(01)\rangle \\
 &\quad + |10\rangle|0 \oplus f(10)\rangle - |10\rangle|1 \oplus f(10)\rangle \\
 &\quad + |11\rangle|0 \oplus f(11)\rangle - |11\rangle|1 \oplus f(11)\rangle \\
 &= \left((-1)^{f(00)}|00\rangle + (-1)^{f(01)}|01\rangle + (-1)^{f(10)}|10\rangle + (-1)^{f(11)}|11\rangle \right) (|0\rangle - |1\rangle)
 \end{aligned}$$

$$|\psi_3\rangle = \begin{pmatrix} (-1)^{f(00)}|+\rangle|+\rangle \\ + \\ (-1)^{f(01)}|+\rangle|-\rangle \\ + \\ (-1)^{f(10)}|-\rangle|+\rangle \\ + \\ (-1)^{f(11)}|-\rangle|-\rangle \end{pmatrix} (|0\rangle - |1\rangle) = \begin{pmatrix} (-1)^{f(00)}(|00\rangle + |01\rangle + |10\rangle + |11\rangle) \\ + \\ (-1)^{f(01)}(|00\rangle - |01\rangle + |10\rangle - |11\rangle) \\ + \\ (-1)^{f(10)}(|00\rangle + |01\rangle - |10\rangle - |11\rangle) \\ + \\ (-1)^{f(11)}(|00\rangle - |01\rangle - |10\rangle + |11\rangle) \end{pmatrix} (|0\rangle - |1\rangle)$$



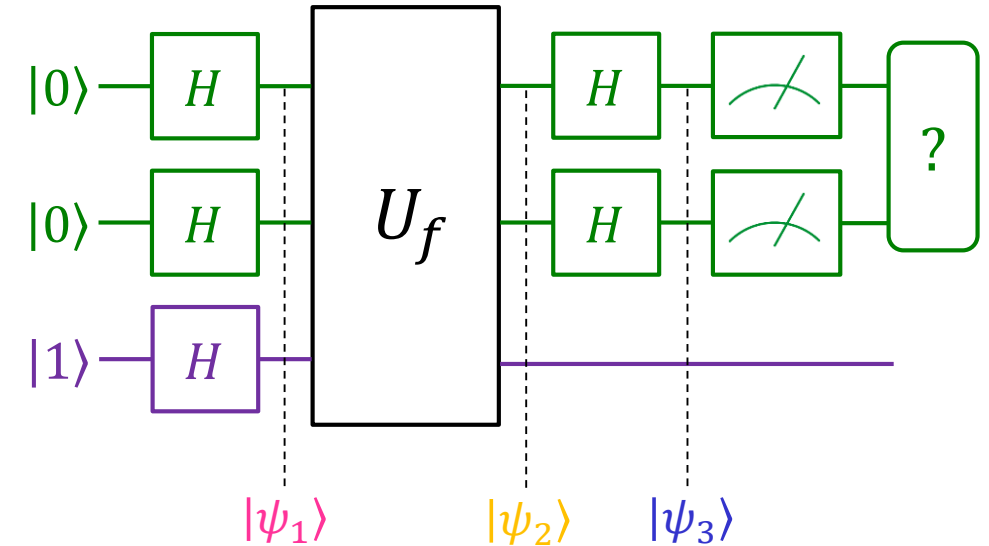
Case 1: $f(\mathbf{x})$ constant

The Deutsch–Jozsa's algorithm

$$\begin{aligned}
 |\psi_1\rangle &= |+\rangle|+\rangle|-\rangle = (|0\rangle + |1\rangle)(|0\rangle + |1\rangle)(|0\rangle - |1\rangle) \\
 &= |00\rangle|0\rangle + |01\rangle|0\rangle + |10\rangle|0\rangle + |11\rangle|0\rangle \\
 &\quad - |00\rangle|1\rangle - |01\rangle|1\rangle - |10\rangle|1\rangle - |11\rangle|1\rangle
 \end{aligned}$$

$$\begin{aligned}
 |\psi_2\rangle &= |00\rangle|0 \oplus f(00)\rangle - |00\rangle|1 \oplus f(00)\rangle \\
 &\quad + |01\rangle|0 \oplus f(01)\rangle - |01\rangle|1 \oplus f(01)\rangle \\
 &\quad + |10\rangle|0 \oplus f(10)\rangle - |10\rangle|1 \oplus f(10)\rangle \\
 &\quad + |11\rangle|0 \oplus f(11)\rangle - |11\rangle|1 \oplus f(11)\rangle \\
 &= \left((-1)^{f(00)}|00\rangle + (-1)^{f(01)}|01\rangle + (-1)^{f(10)}|10\rangle + (-1)^{f(11)}|11\rangle \right) (|0\rangle - |1\rangle)
 \end{aligned}$$

$$|\psi_3\rangle = \begin{pmatrix} (-1)^{f(00)}|+\rangle|+\rangle \\ + \\ (-1)^{f(01)}|+\rangle|-\rangle \\ + \\ (-1)^{f(10)}|-\rangle|+\rangle \\ + \\ (-1)^{f(11)}|-\rangle|-\rangle \end{pmatrix} (|0\rangle - |1\rangle) = \begin{pmatrix} (-1)^{f(00)}(\cancel{|00\rangle} + |01\rangle + |10\rangle + |11\rangle) \\ + \\ (-1)^{f(01)}(\cancel{|00\rangle} - |01\rangle + |10\rangle - |11\rangle) \\ + \\ (-1)^{f(10)}(\cancel{|00\rangle} + |01\rangle - |10\rangle - |11\rangle) \\ + \\ (-1)^{f(11)}(\cancel{|00\rangle} - |01\rangle - |10\rangle + |11\rangle) \end{pmatrix} (|0\rangle - |1\rangle)$$

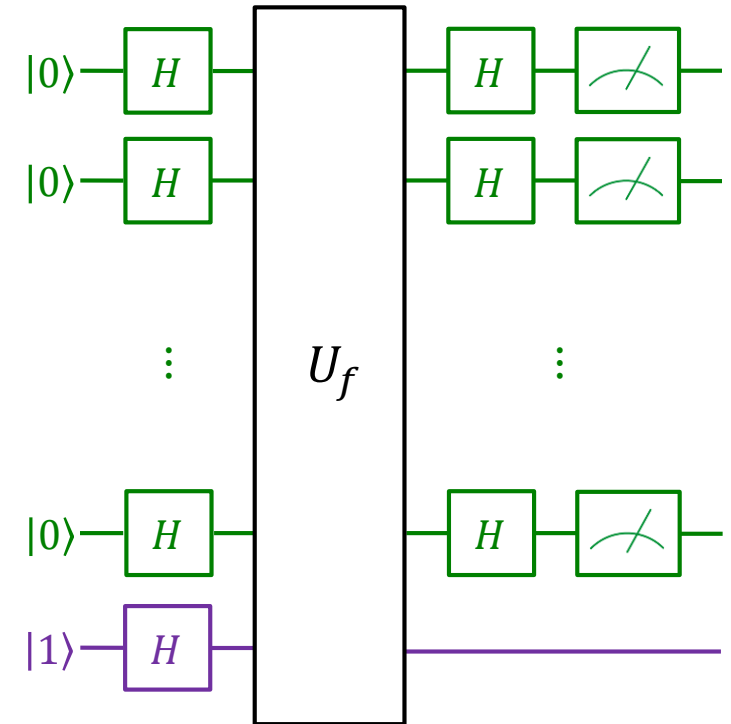


Case 2: $f(\mathbf{x})$ balanced

$\mathbf{x}$	$f(\mathbf{x})$
00	0
01	0
10	1
11	1

The Deutsch–Jozsa's algorithm

- Deutsch–Jozsa' algorithm distinguishes constant and balanced $f(\mathbf{x})$ with **one** call to U_f
- Easily generalized to functions of n bits
- Classical algorithm must make $2^n/2 + 1$ calls to f in the worst-case



RSA

3201909566633586449214513380673586939450873286399666094267120718939118274764109745096207
6125135854709230726844789495069982466049388870402118423844850247561515056606700311858242
0518662612466035933310806051523543514477240831745079737294670568484059288893966801597191
467365490104012805613275753498397059

×

9885364507897925423413334435691578031445015223779047525497121120289956756462872324091661
5573358213902230337405496199685239903992607947536237079716675815466290317162811657581034
8144067758491070167747731689497143989938197783034752714863146991481818790219900682221235
867856812523290666973816995089736641

=

3165204318749848296253197788486522465067406603239583600353846010912505853471662991100934
1429658882895178681199387920863705605216509425059704209474199470560164828053177481317157
0438009992252502156555181960661954284478375382395596097508987165694925972735077887641373
4816870435117749450724048882900685534692391108922744941745280304064669188414143617646922
5624577397858019429231854304158876794839144096761120165707254078306168899976713653152151
7836263344950998368290805821692317187582847379556514666396432652300306251938195573106640
145521269480569790920294930278371419228836111111442366437907855958938819

Shor's algorithm

Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*

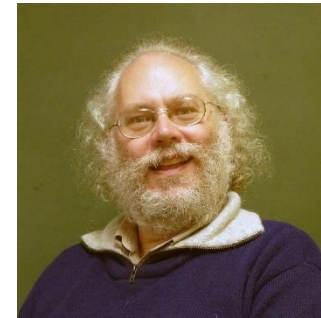
Peter W. Shor[†]

Abstract

A digital computer is generally believed to be an efficient universal computing device; that is, it is believed able to simulate any physical computing device with an increase in computation time by at most a polynomial factor. This may not be true when quantum mechanics is taken into consideration. This paper considers factoring integers and finding discrete logarithms, two problems which are generally thought to be hard on a classical computer and which have been used as the basis of several proposed cryptosystems. Efficient randomized algorithms are given for these two problems on a hypothetical quantum computer. These algorithms take a number of steps polynomial in the input size, e.g., the number of digits of the integer to be factored.

Keywords: algorithmic number theory, prime factorization, discrete logarithms, Church's thesis, quantum computers, foundations of quantum mechanics, spin systems, Fourier transforms

1994



First: something completely different

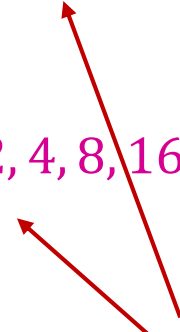
2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, ...

2, 4, 8, 1, 2, 4, 8, 1, 2, 4, 8, 1, 2 ...

mod 15

2, 4, 8, 16, 11, 1, 2, 4, 8, 16, 11, 1, 2 ...

mod 21



sequences are periodic

Factoring to order-finding

$$N = pq$$

$$\underbrace{a^1, a^2, a^3, \dots, a^r}_{=1}, a^1, a^2 \dots \pmod{N}$$

order of a = the smallest positive r such that $a^r = 1 \pmod{N}$

Fact: r must divide $(p-1)(q-1) = \phi(N)$

Proof:

Euler's theorem: for all $a \in \mathbb{Z}_N^*$

$$a^{\phi(N)} = a^{(p-1)(q-1)} = 1 \pmod{N}$$

1. $(p-1)(q-1) = sr + t$ $0 \leq t < r$
2. $a^{(p-1)(q-1)} = a^{sr+t} = a^{sr} a^t = (a^r)^s a^t = 1 \cdot a^t = a^t = 1 \pmod{N} \implies t = 0$ (since r is the *smallest*)
3. $(p-1)(q-1) = sr$ QED

Conclusion: learn $r \implies$ we learn a factor of $(p-1)(q-1)$

repeat with a different $a \implies$ learn another factor of $(p-1)(q-1)$ (with high prob.)

eventually we can learn full $\phi(N) = (p-1)(q-1) \implies$ can find p and q (Problem set 9)

Shor's algorithm

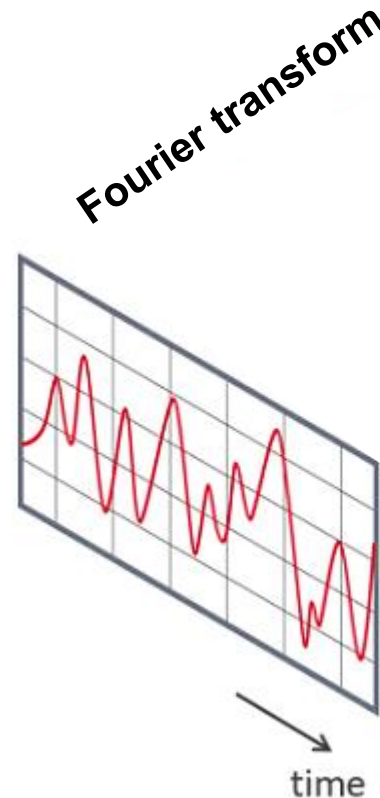
Where the quantum magic happens!

Shor's algorithm	
Input: $N = pq$ Output: p and q	
1.	repeat until $\phi(N)$ is factored:
2.	$a \xleftarrow{\$} \mathbf{Z}_N$
3.	$r \leftarrow \text{Order}_N(a)$ // but how to find r ?
4.	use r to find factor of $\phi(N)$
5.	compute p and q from N and $\phi(N)$

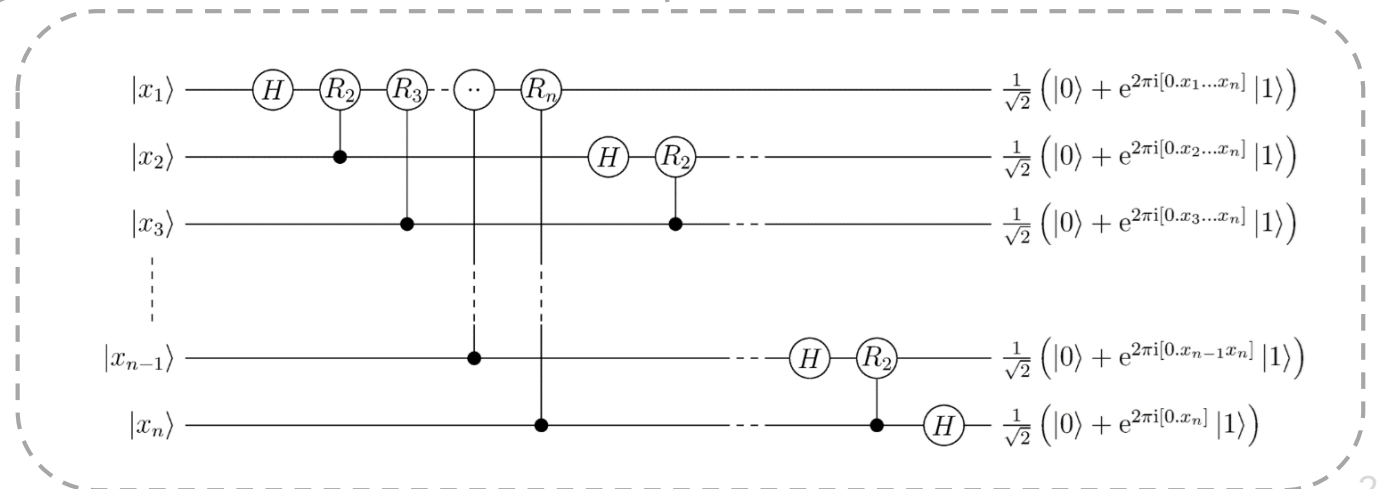
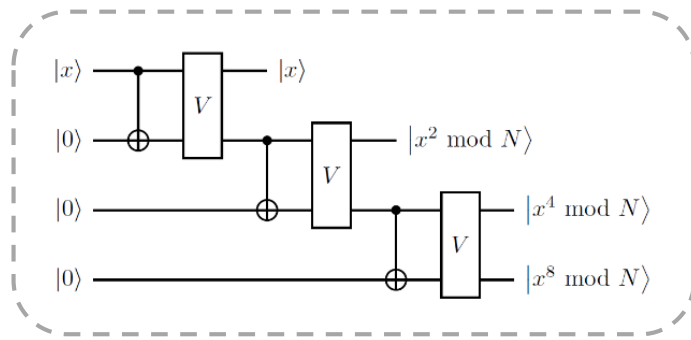
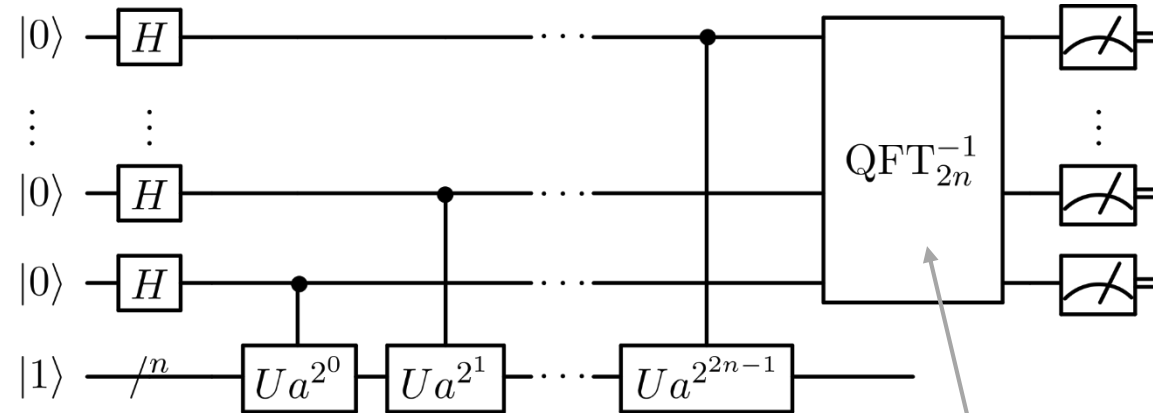
Shor's algorithm

- To factor N : find order r of a in $\mathbb{Z}_N^*$
- Problem: r can be very large
 - Classical solutions take exponential time
- **Note:** the function $f(i) = a^i \bmod N$ is *periodic*:
$$f(i + kr) = a^{i+kr} \bmod N = a^i \bmod N = f(i)$$
 - finding signal frequencies $\Leftrightarrow$ finding signal period
- Key ingredient of Shor's algorithm:

quantum Fourier transform (QFT)

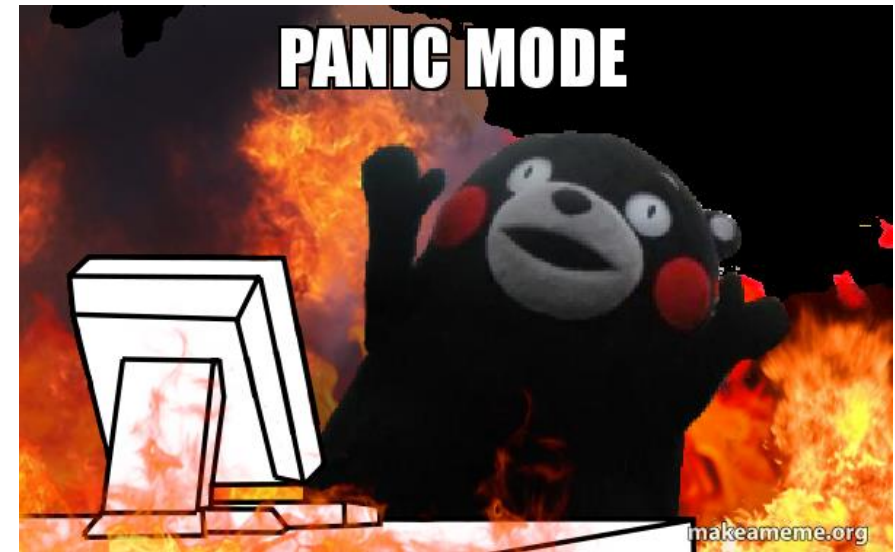


Shor's algorithm

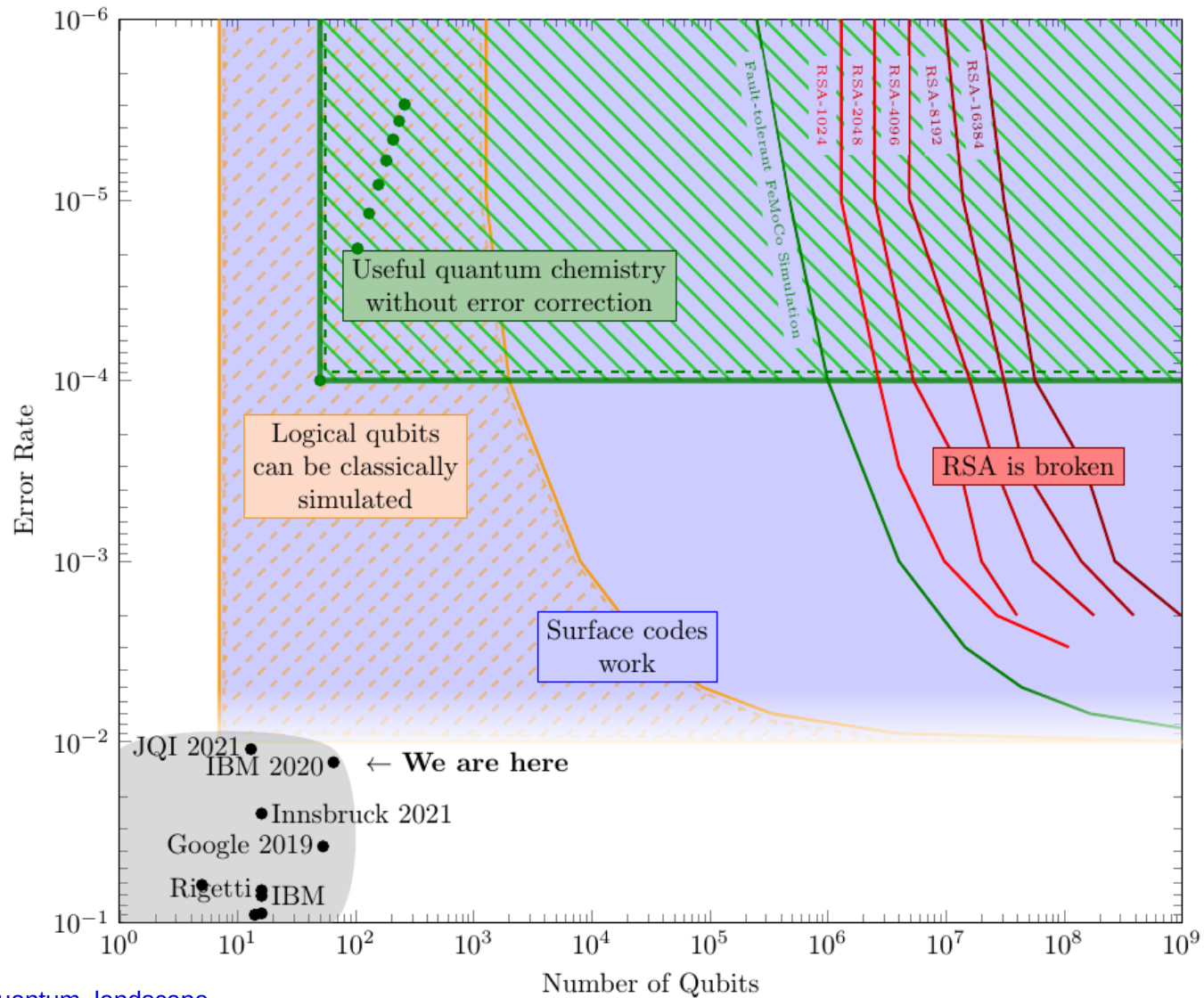


The quantum menace

- ~~RSA~~ – broken by Shor's algorithm
- ~~Diffie-Hellman~~ – broken by Shor's algorithm
- ~~ECDSA~~ – broken by Shor's algorithm
- ...public-key cryptography is basically dead

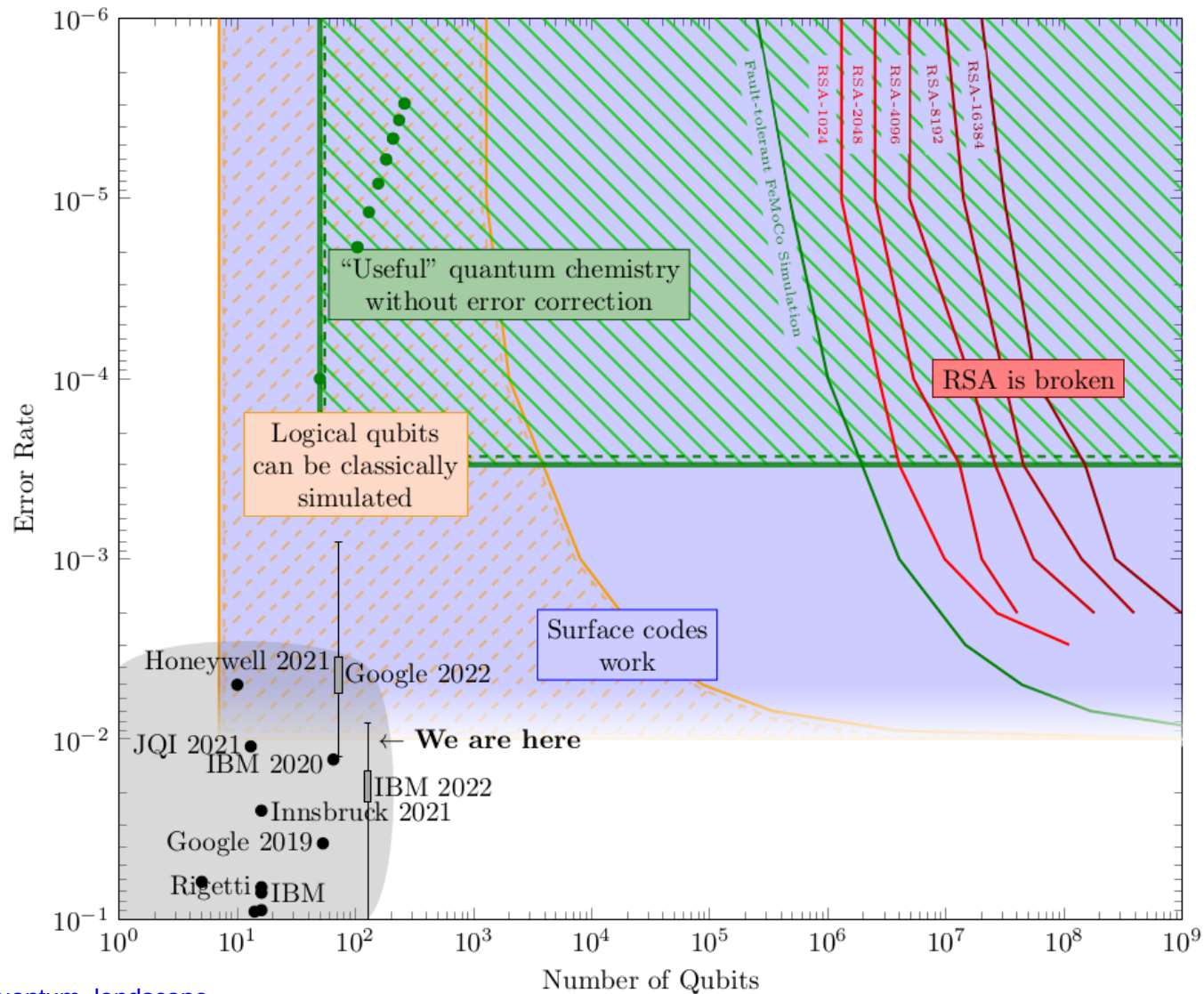


How far away are we?



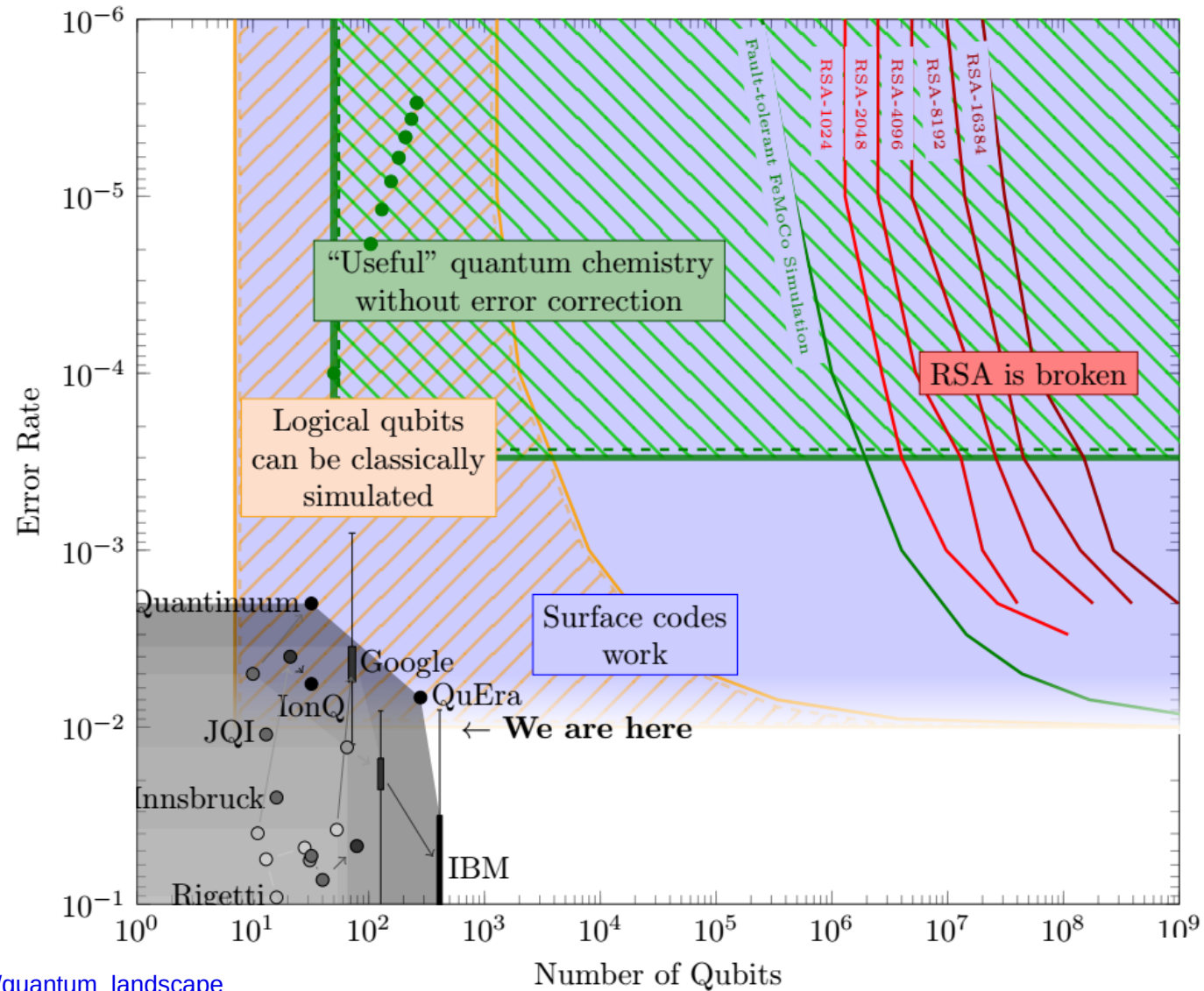
2021

How far away are we?



2022

How far away are we?



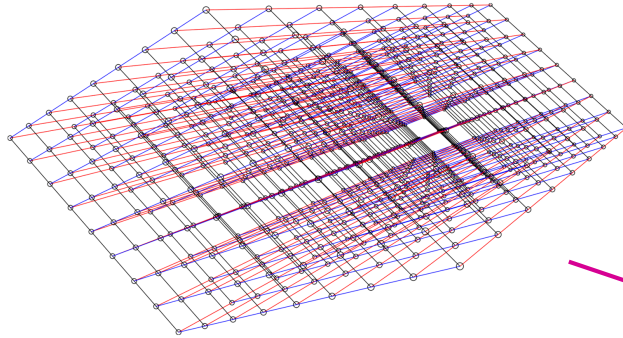
2023

Dealing with quantum computers

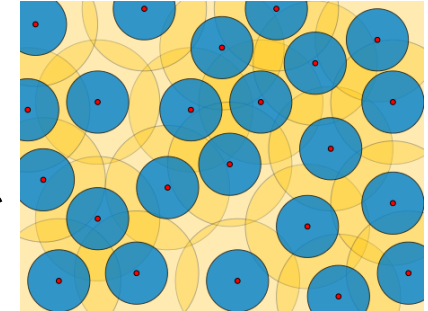
- **Symmetric cryptography**
 - Grover's algorithm: solves $\mathcal{O}(2^n)$ problems in $\mathcal{O}(2^{n/2})$ quantum steps
 - Inherently serial + huge constants
 - AES-128 is most likely safe; using AES-256 removes any doubts
- **Quantum cryptography**
 - Use quantum mechanics to build cryptography
 - Requires specialized equipment
 - Only used for key distribution; does not solve authentication problem
- **Post-quantum cryptography**
 - Classical (asymmetric) algorithms believed to withstand quantum attacks

Post-quantum cryptography

Lattice-based cryptography



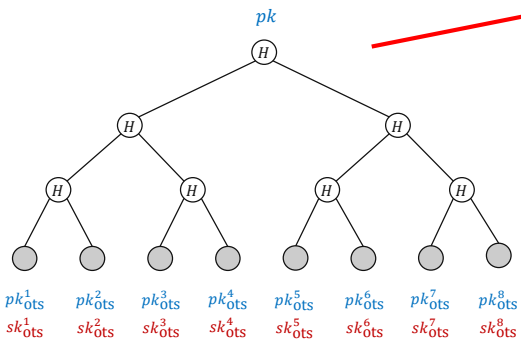
Code-based encryption



<https://eprint.iacr.org/2022/975>



Supersingular isogeny key exchange



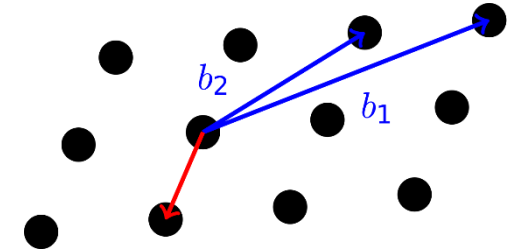
Hash-based signatures



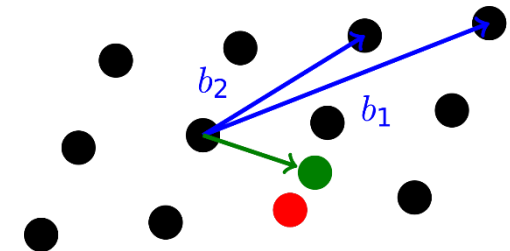
Lattice-based cryptography

- Very versatile computational problems
 - Public-key encryption
 - Digital signatures
 - Hash functions
 - Fully homomorphic encryption
 - Key exchange
- Based on hardness of problems in lattices
 - Believed to be hard for quantum computers too!

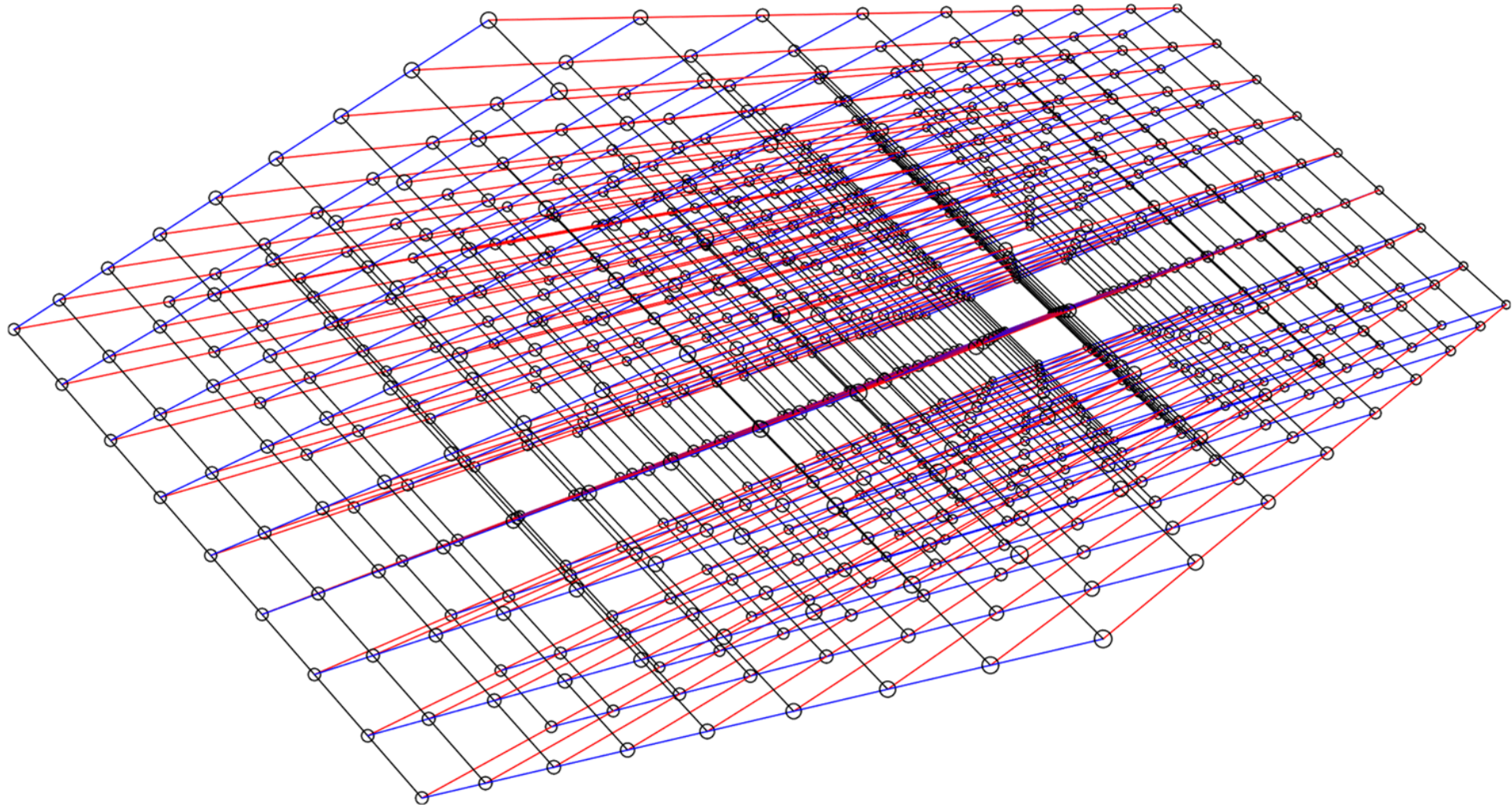
Shortest vector problem

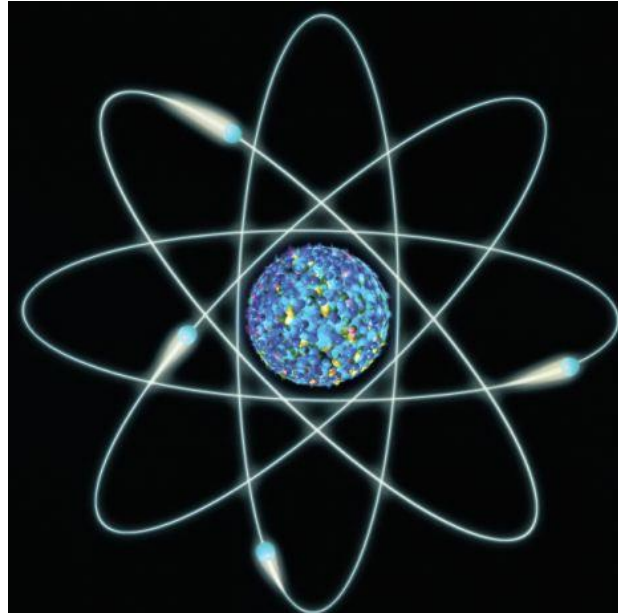


Closest vector problem



Lattice-based cryptography





Questions?