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

TEK4500

16.11.2022

Håkon Jacobsen

hakon.jacobsen@its.uio.no

Quantum computing – the starting point

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

Simulating Physics with Computers

Richard P. Feynman

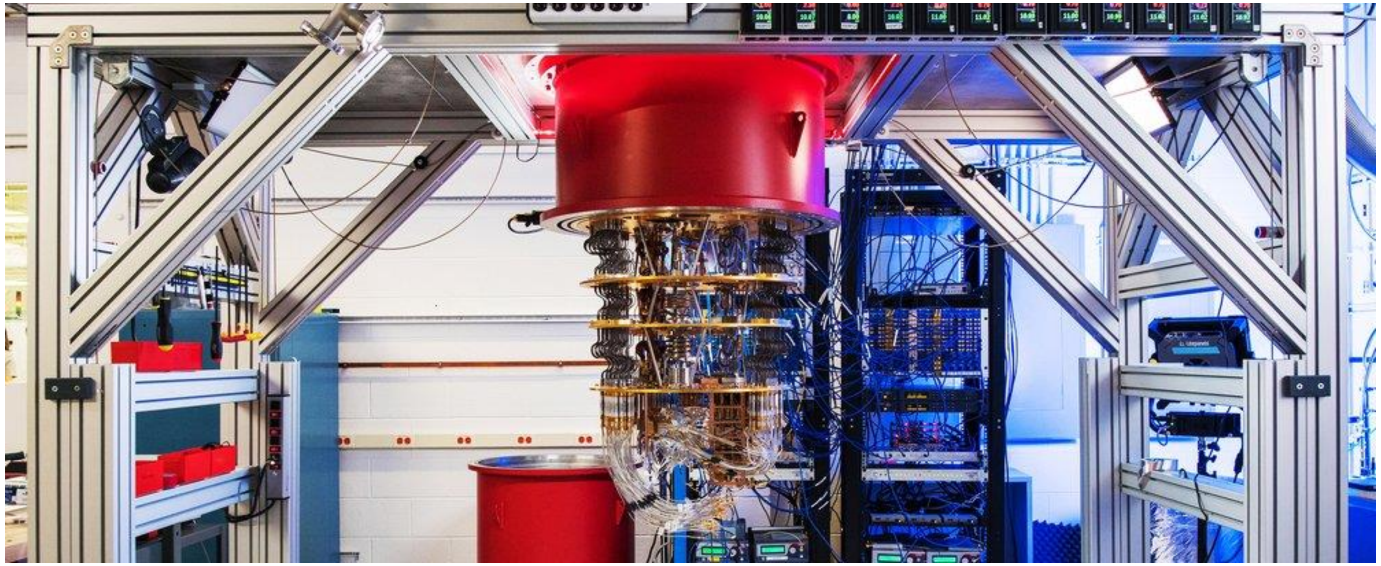
Department of Physics, California Institute of Technology, Pasadena, California 91107

Received May 7, 1981

Elements of (quantum) computing

- Three elements of all computations: data, operations, results

- Quantum computation
  - Data = **qubit**
  - Operation = **quantum gate**
  - Results = **measurements**



Qubits

- Classical bit:

$0 \qquad\qquad 1$

- Qubit:

  Can be in a **superposition** of two basic states $|0\rangle$ and $|1\rangle$

  $$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \qquad\qquad \alpha, \beta \in \boldsymbol{C} \qquad |\alpha|^2 + |\beta|^2 = 1$$

  But we can never observe α and β directly!

  Must **measure** $|\psi\rangle$ to obtain its value $\Longrightarrow$ state *randomly* collapses to either $|0\rangle$ or $|1\rangle$

  What's the probability of observing $|0\rangle$ or $|1\rangle$?

  $$\Pr[\text{observe } |0\rangle] = |\alpha|^2$$

  $$\Pr[\text{observe } |1\rangle] = |\beta|^2$$

Multiple qubits

- 2-qubit system

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

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

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

- N-qubit system: 2^N basis states

$$|\psi\rangle = \sum_{i=0}^{2^N-1} \alpha_i |i\rangle$$

$|\alpha_0|^2 + |\alpha_1|^2 + \cdots + |\alpha_{2^N-1}|^2 = 1$

$$0.8|001\rangle - 0.6i|101\rangle = \begin{pmatrix} 0 \\ 0.8 \\ 0 \\ 0 \\ 0 \\ -0.6i \\ 0 \\ 0 \end{pmatrix}$$

$\Pr[\text{observe } |001\rangle] = |0.8|^2 = 0.64$

$\Pr[\text{observe } |101\rangle] = |-0.6i|^2 = \sqrt{(-0.6)^2}^2 = 0.36$

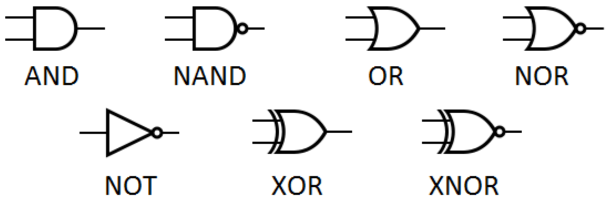
  - Representable by a 2^N element vector:

$$|\psi\rangle = \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_{2^N-1} \end{pmatrix}$$

$|000\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$ $|001\rangle = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$ $|010\rangle = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$ $|011\rangle = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$ $|100\rangle = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$ $|101\rangle = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$ $|110\rangle = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$ $|111\rangle = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$

Quantum computation – quantum gates

- Classic bits are transformed using logical gates



- Qubits are transformed using **quantum gates**

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \overset{G}{\mapsto} |\psi'\rangle = \alpha'|0\rangle + \beta'|1\rangle$$

Operator	Gate(s)	Matrix
Pauli-X (X)	X, $\oplus$	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
Pauli-Z (Z)	Z	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
Hadamard (H)	H	$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
Controlled Not (CNOT, CX)		$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$

(Quantum) NOT-gate (or X gate)

$$|0\rangle \overset{X}{\mapsto} |1\rangle$$

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

$$|1\rangle \overset{X}{\mapsto} |0\rangle$$

X gate:

$$\boldsymbol{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$|\psi\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$$

$$\alpha|0\rangle + \beta|1\rangle \overset{X}{\mapsto} \beta|0\rangle + \alpha|1\rangle$$

$$\boldsymbol{X}|0\rangle = \boldsymbol{X}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle \qquad \boldsymbol{X}|1\rangle = \boldsymbol{X}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle \qquad \boldsymbol{X}|\psi\rangle = \boldsymbol{X}\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \ ?$$

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \qquad \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \beta \\ \alpha \end{pmatrix}$$

The Hadamard gate

$$|0\rangle \overset{H}{\mapsto} \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

$$|1\rangle \overset{H}{\mapsto} \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$$

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

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

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

H gate:

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

$$\Pr[\text{measure } \boldsymbol{H}|0\rangle \Rightarrow |0\rangle] = \left|\frac{1}{\sqrt{2}}\right|^2 = 0.5$$

$$\Pr[\text{measure } \boldsymbol{H}|1\rangle \Rightarrow |1\rangle] = \left|\frac{1}{\sqrt{2}}\right|^2 = 0.5$$

The Hadamard gate allows us to create random bits!

$$\begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix}$$

$$\begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{pmatrix}$$

Controlled-NOT gate (CNOT)

CNOT

$|00\rangle \mapsto |00\rangle$

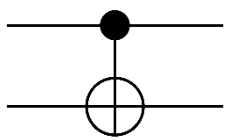
$|01\rangle \mapsto |01\rangle$

$|10\rangle \mapsto |11\rangle$

$|11\rangle \mapsto |10\rangle$

CNOT gate:

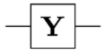
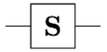
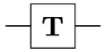
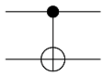
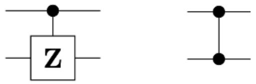
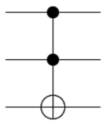
$$\mathbf{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$



$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \overset{|10\rangle}{\begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}} = \overset{|11\rangle}{\begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \delta \end{pmatrix} = \begin{pmatrix} \alpha \\ \beta \\ \delta \\ \gamma \end{pmatrix}$$

Many other 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)		$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$

Universal for classical logic!

Quantum gates

- Turns out that all quantum gates can be described by matrices
  - In fact, very special matrices: unitary matrices
  - … and *only* unitary matrices! (fact of nature)

$$\boldsymbol{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \qquad |0\rangle \mapsto |1\rangle \quad |1\rangle \mapsto |0\rangle$$

- Quantum operations are *linear* and can be combined

$$|\psi_0\rangle \overset{\boldsymbol{Z}}{\mapsto} |\psi_1\rangle \overset{\boldsymbol{X}}{\mapsto} |\psi_2\rangle \overset{\boldsymbol{H}}{\mapsto} |\psi_3\rangle \overset{\boldsymbol{Z}}{\mapsto} |\psi_4\rangle$$

$$\boldsymbol{Z} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \qquad |0\rangle \mapsto |0\rangle \quad |1\rangle \mapsto -|1\rangle$$

$$\boldsymbol{ZHXZ}|\psi_0\rangle = |\psi_4\rangle$$

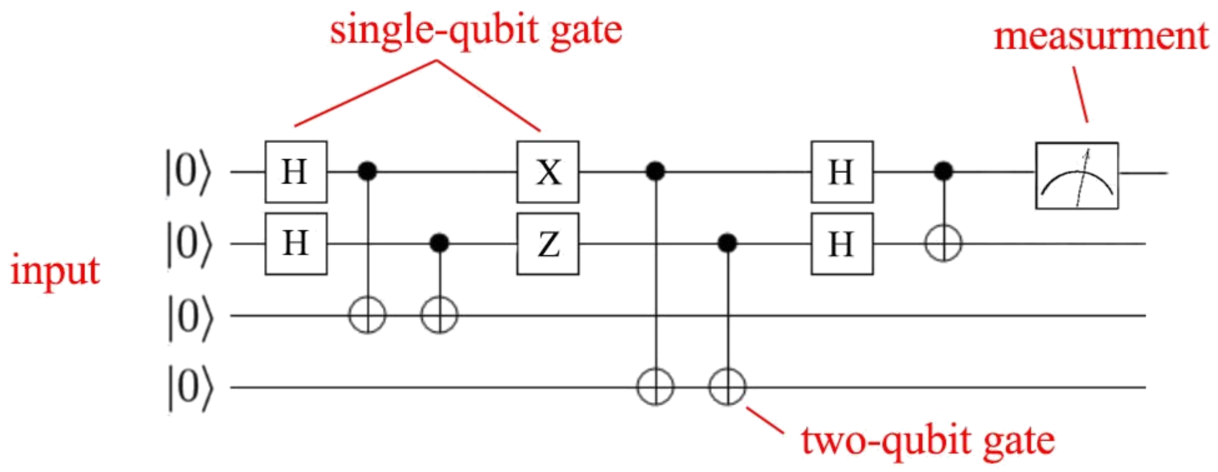
$$\boldsymbol{ZHXZ}|0\rangle = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$= \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

$$\boldsymbol{H} = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} \qquad |0\rangle \mapsto \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \quad |1\rangle \mapsto \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$$

Quantum computer

- A **quantum computer** consists of:
  - N input qubits
  - a sequence of quantum gates
  - N output qubits
  - result = measurement of final quantum state (output qubits)

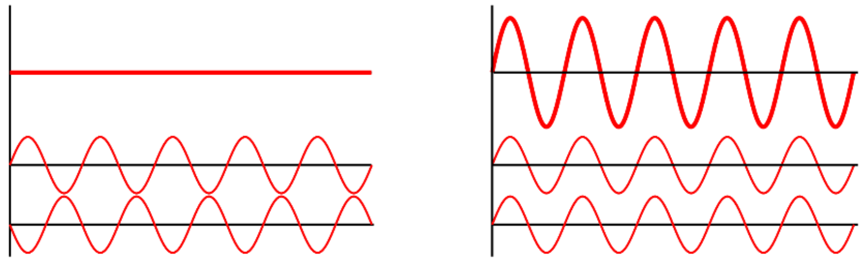


What makes quantum computation special?

- **Warning:** a quantum computer does *not* simply "try out all solutions in parallel"
- The magic comes from allowing *complex* amplitudes (or even just negative reals)

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \qquad \alpha, \beta \in \boldsymbol{C}$$

- **Quantum interference:** can *carefully* choreograph computations so wrong answers "cancel out" their amplitudes, while correct answers "combine"



  - increases probability of measuring correct result
  - only a few special problems allow this choreography

https://www.smbc-comics.com/comic/the-talk-3

Shor's algorithm

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

Peter W. Shor†

1994

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

First: something completely different

$$2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, \ldots$$

$$2, 4, 8, 1, 2, 4, 8, 1, 2, 4, 8, 1, 2 \ldots \qquad \bmod 15$$

$$2, 4, 8, 16, 11, 1, 2, 4, 8, 16, 11, 1, 2 \ldots \qquad \bmod 21$$

sequences are periodic

Factoring to order-finding

$N = pq$

$$\underbrace{a^1, a^2, a^3, \ldots, a^r}_{}\!\!\overset{=1}{}, a^1, a^2 \ldots \quad (\mathrm{mod}\, N)$$

order of a = the smallest positive r such that $a^r = 1 \ (\mathrm{mod}\, N)$

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

$$a^{\phi(N)} = a^{(p-1)(q-1)} = 1 \ (\mathrm{mod}\, N)$$

Fact: r must divide $(p-1)(q-1)$

Proof:

- $(p-1)(q-1) = sr + t \qquad 0 \le t < r$
- $a^{(p-1)(q-1)} = a^{sr+t} = a^{sr}a^t = (a^r)^s a^t = 1 \cdot a^t = a^t = 1 \bmod N \quad \Longrightarrow \quad t = 0$ (since r is the *smallest*)
- $(p-1)(q-1) = sr$

QED

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

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

eventually we can learn full $(p-1)(q-1) \implies$ 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. $a \xleftarrow{\$} \boldsymbol{Z}_N$
2. $r \leftarrow \mathrm{Order}_N(a)$ // but how to find r?
3. use r to find $\phi(N)$
4. compute p and q from N and $\phi(N)$

Shor’s algorithm

- To factor N: find order r of a in $\mathbf{Z}_N^*$

- Problem: r can be very large
  - Classical solutions take exponential time

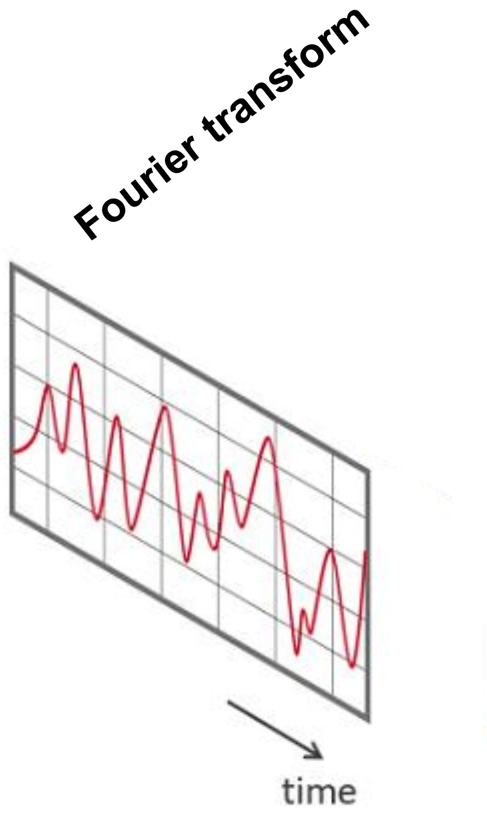
- **Note:** the function $f(i) = a^i \bmod N$ is *periodic*:

$$f(i+kr) = a^{i+kr} = a^i \bmod N = f(i)$$

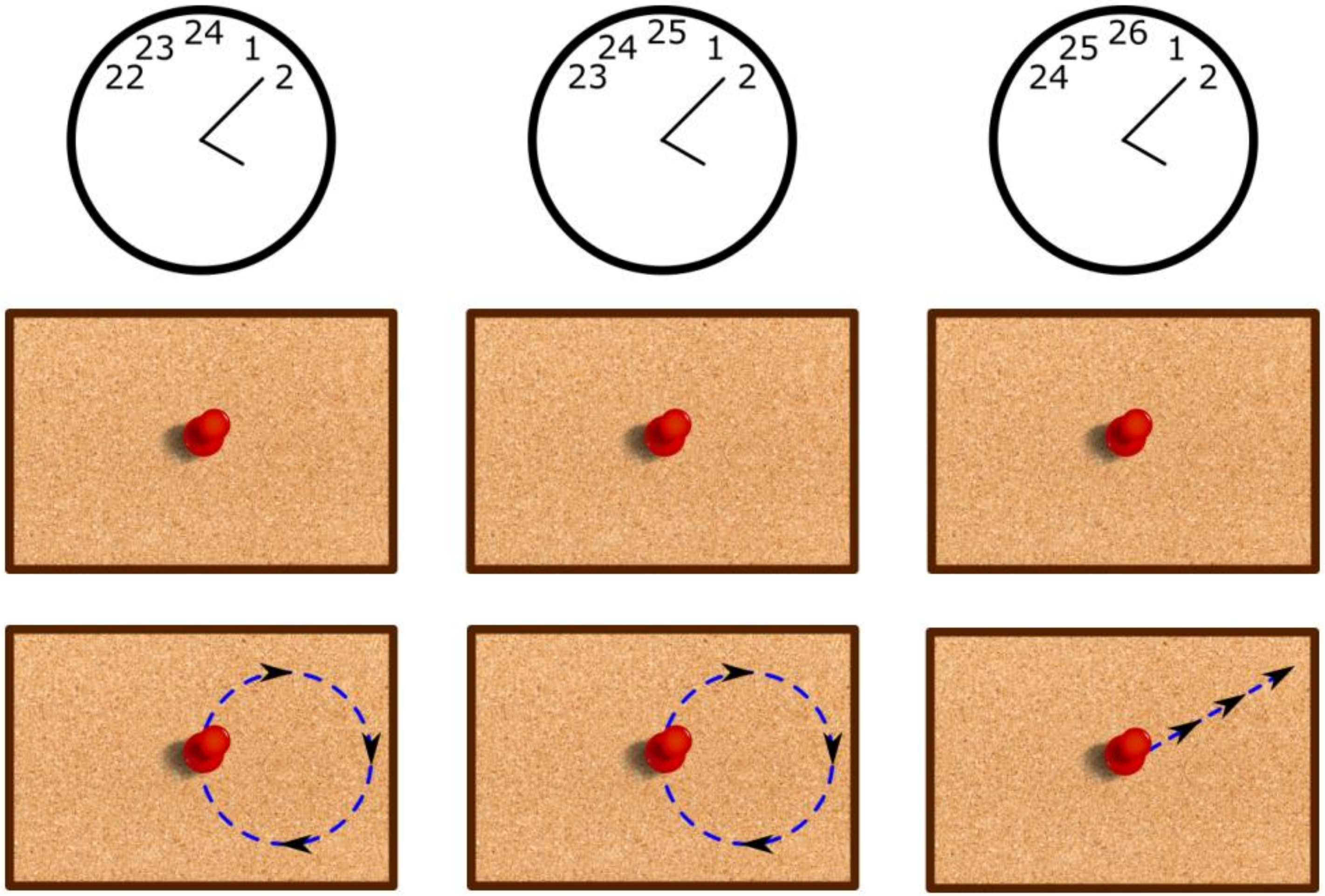
  - finding signal frequencies ⟺ finding signal period

- Key ingredient of Shor's algorithm:

quantum Fourier transform (QFT)



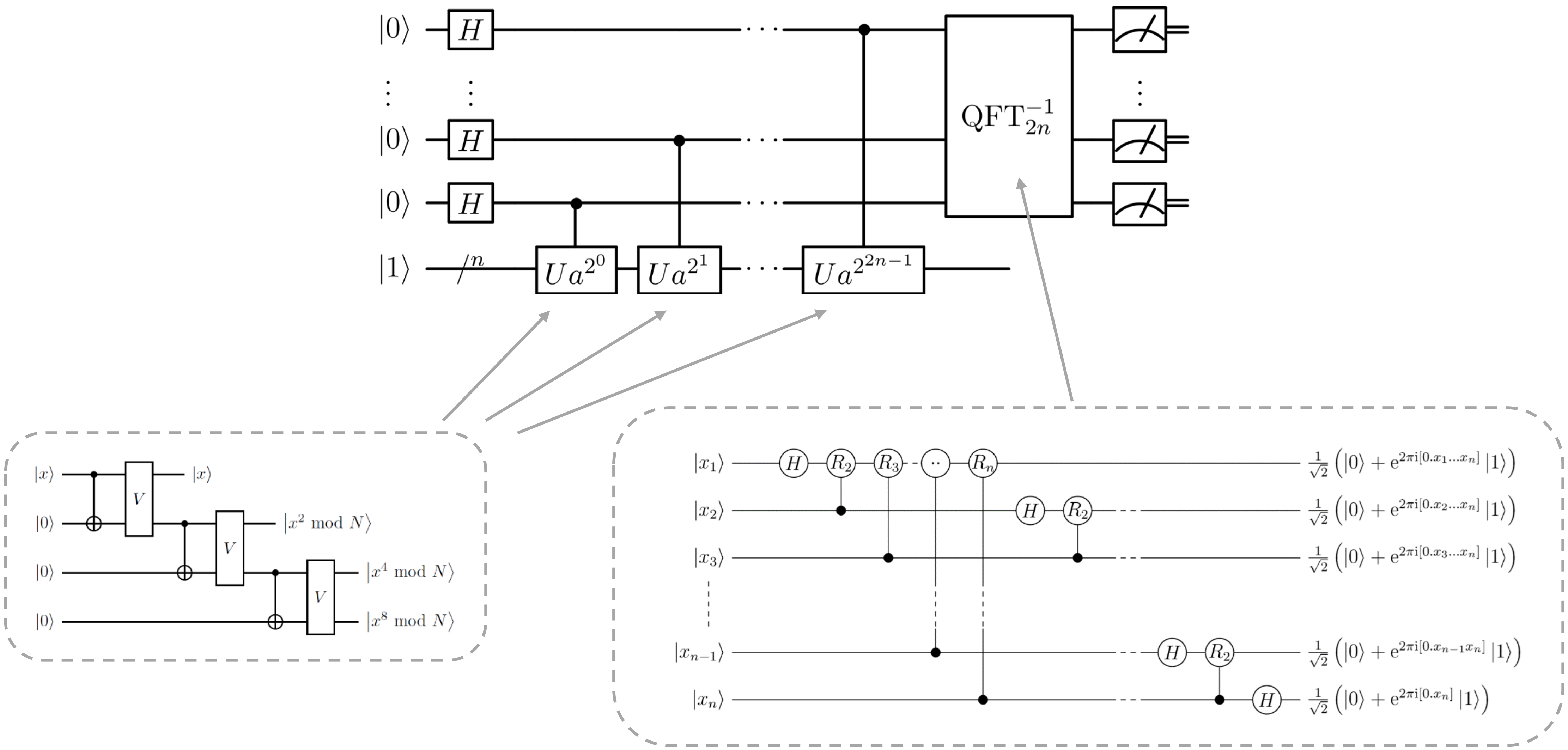
Fourier transform



Source: https://www.scottaaronson.com/qclec.pdf

More on the Fourier transform:
(3Blue1Brown) https://www.3blue1brown.com/lessons/fourier-transforms
(Veritasium) https://youtu.be/nmgFG7PUHfo

Shor's algorithm



Consequences of Shor's algorithm

- Cryptosystems broken by Shors' algorithm:
  - RSA
  - Diffie-Hellman
  - Schnorr
  - ElGamal
  - ECDSA

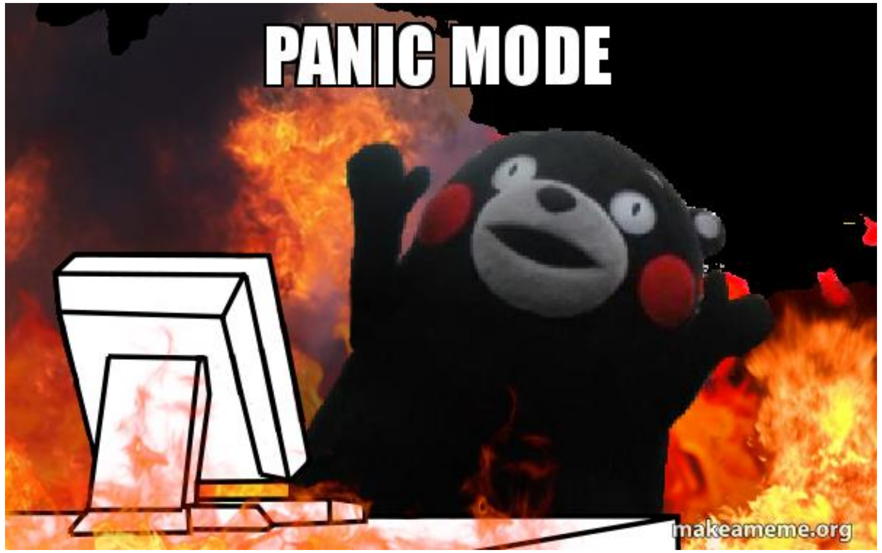
both $\boldsymbol{Z}_p^*$ and $E(\boldsymbol{F}_p)$

- …public-key crypto is dead

Shor's algorithm
Input: $N = pq$ **Output:** p and q 1. $a \xleftarrow{\$} \boldsymbol{Z}_N$ 2. $r \leftarrow \mathrm{Order}_N(a)$ // QFT++ 3. use r to find $\phi(N)$ 4. compute p and q from N and $\phi(N)$

The quantum menace

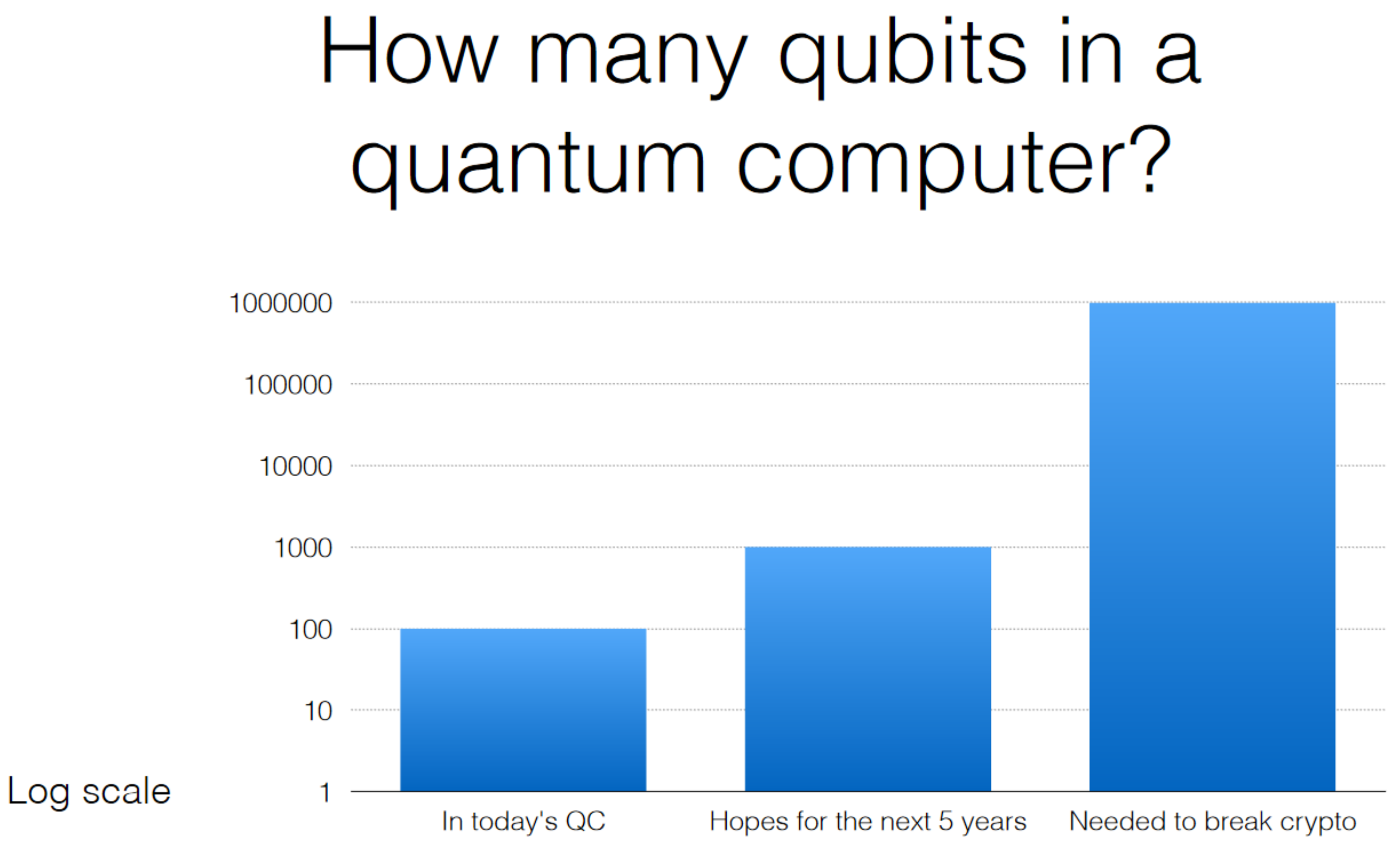
- How far away is a quantum computer?
  - Nobody knows



- Building a large-scale quantum computer is a huge engineering challenge
  - *very* susceptible to noise (decoherence)
  - requires quantum error correction (is it even possible?)
  - many *physical* qubits needed to simulate a single *logical* qubit
    - ≥ 1000 logical qubits needed for Shor's algorithm
    - largest (known) quantum computers:

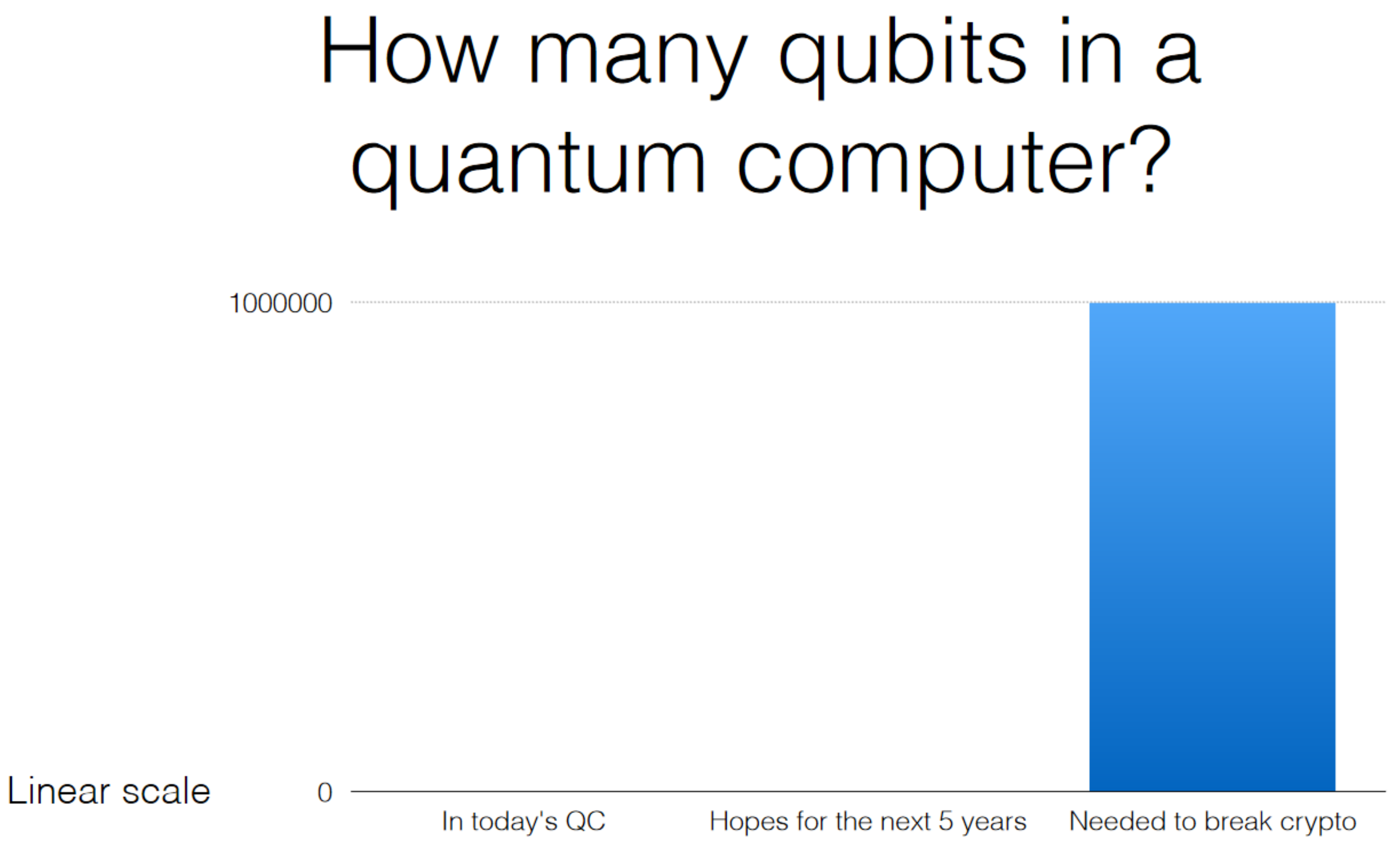
≈ 53 physical qubits (Google; 2019)	(no error correction)
≈ 65 physical qubits (IBM; 2020)	(no error correction)
≈ 127 physical qubits (IBM; 2021)	(no error correction)
≈ 433 physical qubits (IBM; 2022)	(no error correction)

The quantum menace



Source: JP Aumasson; https://www.aumasson.jp/data/talks/quantum-cyberpeace-2021.pdf

The quantum menace



Source: JP Aumasson; https://www.aumasson.jp/data/talks/quantum-cyberpeace-2021.pdf

MIT
Technology
Review

Topics+ Top Stories Maga

Computing

NSA Says It “Must Act Now” Against the Quantum Computing Threat

The National Security Agency is worried that quantum computers will neutralize our best encryption – but doesn’t yet know what to do about that problem.

by Tom Simonite February 3, 2016

Dealing with quantum computers

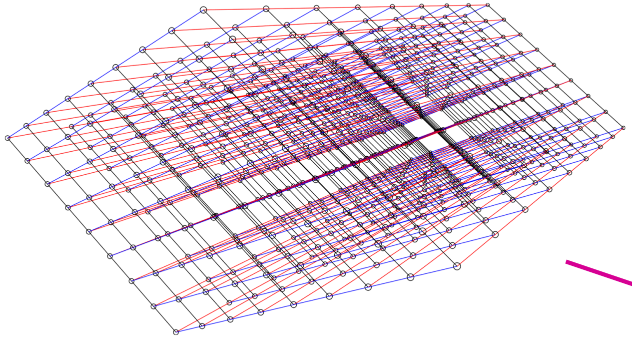
- **Symmetric cryptography**
  - Grover's algorithm: solves $\mathcal{O}(2^n)$ problems in $\mathcal{O}\left(2^{n/2}\right)$ quantum steps
  - Inherently serial + huge constants
  - AES-128 is most likely safe

- **Quantum cryptography**
  - Use quantum mechanics to build cryptography

- **Post-quantum cryptography**
  - Classical algorithms believed to withstand quantum attacks

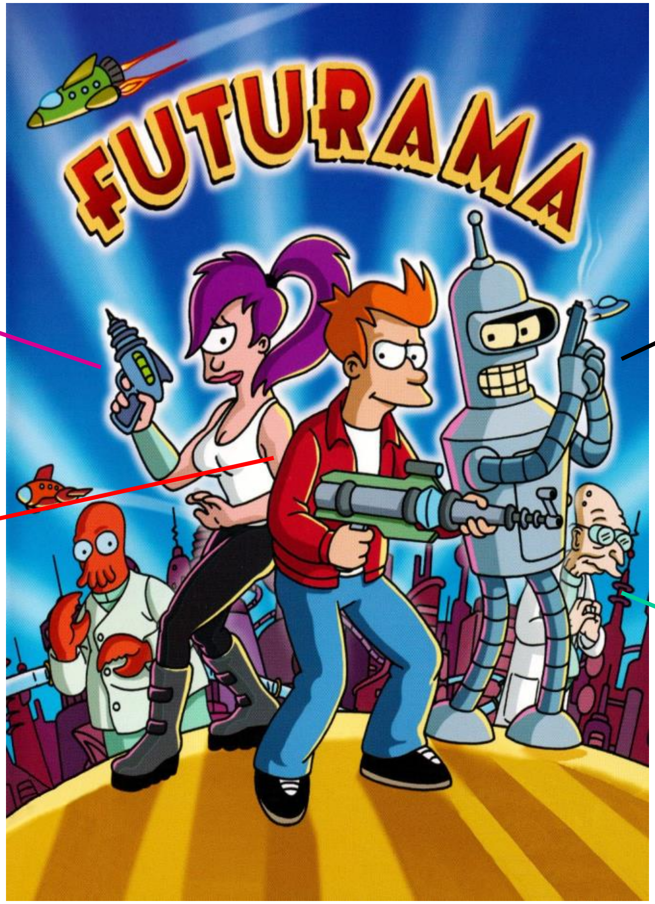
Post-quantum cryptography

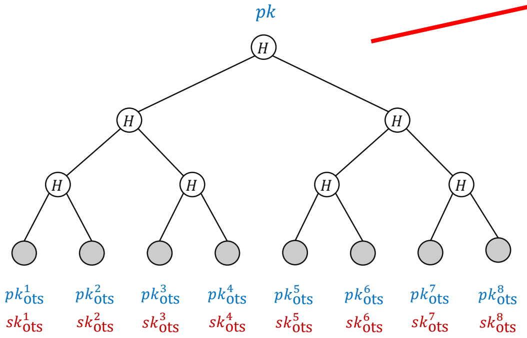
Lattice-based cryptography



Code-based encryption

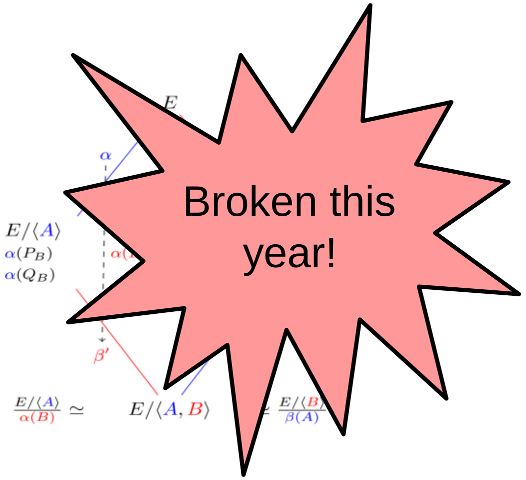
$$c = Gx + e \in F_2^n$$





Hash-based signatures

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



Supersingular isogeny key exchange

The NIST post-quantum competition

- Public competition to standardize post-quantum schemes
  - Public-key encryption
  - Digital signatures

- Started in 2017
  - Round 1: 69 submissions
  - Round 2: 26 candidates selected
  - Round 3: 15 candidates selected

SPHINCS+ Hash-based

The NIST post-quantum competition

- Public competition to standardize post-quantum schemes
  - Public-key encryption
  - Digital signatures

- Started in 2017
  - Round 1: 69 submissions
  - Round 2: 26 candidates selected
  - Round 3: 15 candidates selected

- Winners:
  - CRYSTALS-KYBER (PKE)
  - CRYSTALS-DILITHIUM (Signature)
  - Falcon (Signature)
  - SPHINCS+ (Signature)

Algorithm (public-key encryption)	**Problem**
Classic McEliece	Code-based
CRYSTALS-KYBER	Lattice-based
NTRU	Lattice-based
SABER	Lattice-based
BIKE	Code-based
FrodoKEM	Lattice-based
HQC	Code-based
NTRU Prime	Lattice-based
SIKE	Isogeny-based

Algorithm (digital signatures)	**Problem**
CRYSTALS-DILITHIUM	Lattice-based
Falcon	Lattice-based
Rainbow	Multivariate-based
GeMSS	Multivariate-based
Picnic	ZKP
SPHINCS+	Hash-based

The NIST post-quantum competition

- Public competition to standardize post-quantum schemes
  - Public-key encryption
  - Digital signatures

- Started in 2017
  - Round 1: 69 submissions
  - Round 2: 26 candidates selected
  - Round 3: 15 candidates selected
  - Round 4: alternative candidates

- Winners:
  - CRYSTALS-KYBER (PKE)
  - CRYSTALS-DILITHIUM (Signature)
  - Falcon (Signature)
  - SPHINCS+ (Signature)

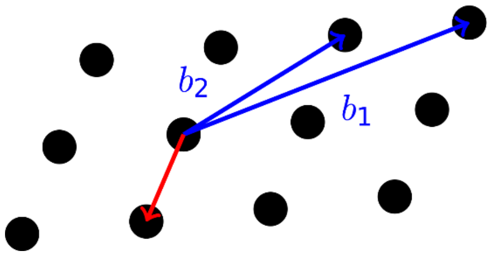
Algorithm (public-key encryption)	Problem
Classic McEliece	Code-based
CRYSTALS-KYBER	Lattice-based
NTRU	Lattice-based
SABER	Lattice-based
BIKE	Code-based
FrodoKEM	Lattice-based
HQC	Code-based
NTRU Prime	Lattice-based
SIKE	Isogeny-based

Algorithm (digital signatures)	Problem
CRYSTALS-DILITHIUM	Lattice-based
Falcon	Lattice-based
Rainbow	Multivariate-based
GeMSS	Multivariate-based
Picnic	ZKP
SPHINCS+	Hash-based

Lattice-based cryptography

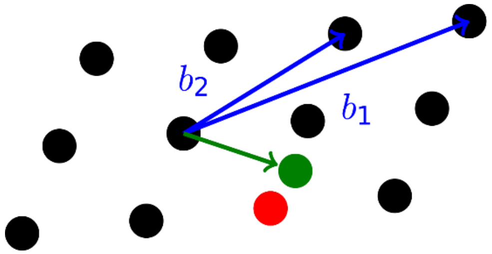
- Very versatile computational problems
  - Public-key encryption
  - Digital signatures
  - Hash functions
  - Fully homomorphic encryption
  - Key exchange

Shortest vector problem



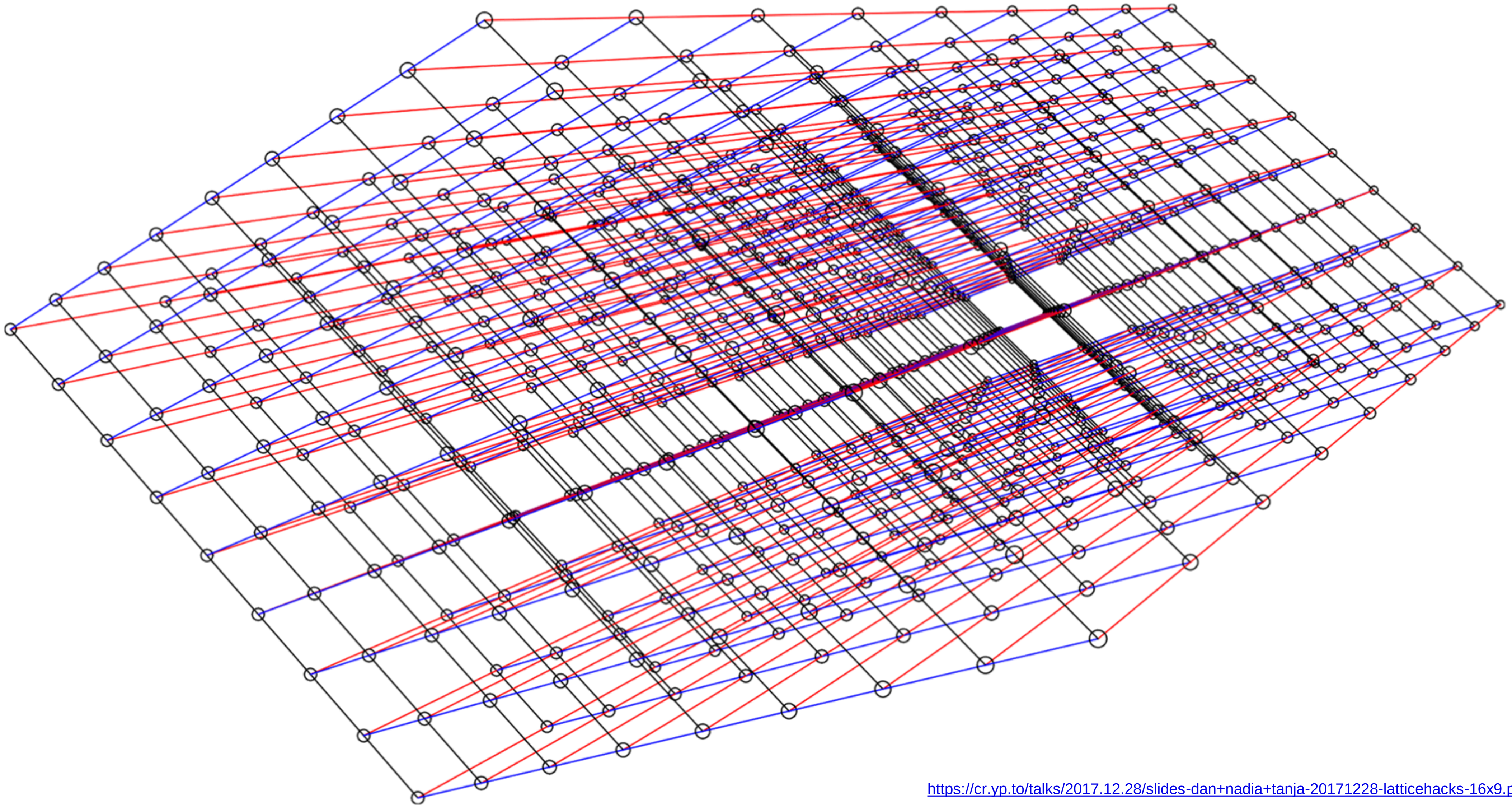
- Leads to efficient and compact schemes

Closest vector problem



- Based on hardness of problems in algebraic number theory
  - Believed to be hard also for quantum computers

Lattice-based cryptography



https://cr.yp.to/talks/2017.12.28/slides-dan+nadia+tanja-20171228-latticehacks-16x9.pdf

Post-quantum cryptography

- Want to learn more about post-quantum cryptography?

- Sign up for TEK5550 - Advanced Topics in Cryptology next spring!

Next week

- Summary lecture
  - If there's anything in particular you want me to repeat, let me know!

- Ask me anything session