Part 5

How does Shor's algorithm work?

- Factoring numbers with Shor’s algorithm begins with selecting a **random integer smaller** than the number to be factored.
- The classically-calculated greatest common divisor (GCD) of these two numbers, the random number and the target number, is then used to determine whether the target number has already been factored accidentally. For smaller numbers, that’s a possibility.
- For larger numbers, a **supercomputer** could be needed. And for numbers that are believed to be cryptographically secure, a **quantum computer** will be needed.

- The role of the quantum computer is to determine the **period** of the number to be factored.
- The calculation results determine whether a new random integer needs to be tested, or whether the sought-after factors have been discovered.
- Once the target integer has been factored, the role of Shor’s Algorithm is concluded

- **How is Shor's algorithm implemented?**
- Shor’s Factoring Algorithm is not simple to implement. First of all, the algorithm has three major components: **one using classical computation, one using quantum computation, and another using classical computation**.

- Adding to this complexity, however, the quantum component of Shor’s Factoring Algorithm has four subcomponents.
- And while two of those quantum subcomponents are relatively straightforward to explain, the other two are incredibly important quantum subroutines.
- In fact, they are arguably the two most significant quantum subroutines.

- One of the critical quantum subcomponents is called **quantum phase estimation** (QPE). The important takeaway here is that it performs the modular arithmetic needed to find the **period** of the number to be factored. In other words, this is where the factoring power of Shor’s algorithm comes from.

- The other key quantum subcomponent is called the **inverse Quantum Fourier Transform** (iQFT). Simply put, the iQFT takes the quantum result of the modular arithmetic that immediately precedes it and transforms it into classical information that can be retrieved from the quantum circuit through a process known as measurement.

- All together, Shor’s Factoring Algorithm begins with a few classical steps. The quantum component then finds the **period of the number** to be factored.
- This is done through quantum modular arithmetic, the result of which is converted from quantum information to classical information so that it is useable.
- And, finally, there are another couple of classical steps. If the answer is not found, and the number consequently cannot be factored, the algorithm in its entirety is adjusted and repeated.

- **How many qubits are needed for Shor's algorithm?**
- A distinction first needs to be made between **physical and logical qubits** to answer this question.
- All present-day qubits are **physical qubits**. They are extremely “noisy,” which means they are **error-prone**.
- The results of quantum computation of any significance make it impossible to distinguish correct answers from incorrect ones.
- Every possible solution has the same probability of being right, which is the same as having no answers at all.

- That’s where **logical qubits** come into play. Physical qubits need to be connected and structured in ways that they will collectively provide enough error correction to each other to be considered “fault-tolerant” collectively. At that point, these qubit collectives become known as “**logical qubits**,” or sometimes “**perfect qubits**.”

- These logical qubits are abstractions. A quantum circuit today with five qubits represents five highly-noisy, error-prone physical qubits.
- Quantum algorithm designers of the near future will want those five qubits to represent logical, fault-tolerant, error-free qubits.
- The estimates vary as to how many physical qubits will be needed to represent one logical qubit, but a reasonable number to work with is **1,000.**
- Most estimate that a quantum computer will need around **1,000 physical qubits** to represent just **one logical qubit**.

- To appreciate how challenging that is, the largest quantum computer today has **only 127 physical qubits**.
- And while IBM has a goal of unveiling a 1,000-qubit device by next year, that is still only 1,000 physical qubits. Much work remains to engineer the first logical qubit.

- Estimates of the number of qubits Shor’s Factoring Algorithm needs vary considerably.
- First, care must be taken, as noted, to discern estimates for logical qubits and estimates for physical qubits. Depending on the researchers' objectives, the number of required qubits can be reduced, **sacrificing time of execution and circuit depth**.
- On the other hand, the number of qubits can be significantly expanded, **reducing runtime and shedding circuit depth**.
- The differences are that lower qubit counts will become available sooner, while the quickest runtimes will be the most advantageous. A selection of estimates follows.

- In an August 1, 1996 paper titled, “Efficient networks for quantum factoring” by David Beckman, Amalavoyal N. Chari, Srikrishna Devabhaktuni, and John Preskill, the authors estimated that factoring a K-bit number would take K^3 time and require **5K+1 logical qubits**.
- Factoring a 2,048-bit number, referring to breaking RSA encryption, would therefore take 8.6 billion time and require 10,241 logical qubits, or roughly *10 million physical qubits*.
- Unfortunately, it’s not clear how long 8.6 billion time would be, as **different qubit technologies operate at different speeds**.

- Then in a February 21, 2003, paper titled, "Circuit for Shor's algorithm using 2n+3 qubits" by St´ephane Beauregard, the authors estimated that factoring a K-bit number would require 2n+3 logical qubits.
- Factoring a 2,048-bit number would therefore require only 4,099 logical qubits, or roughly 4 million physical qubits.
- Again, **zero logical qubits** exist today. However, this paper nonetheless brought Shor's Factoring Algorithm closer to implementation.

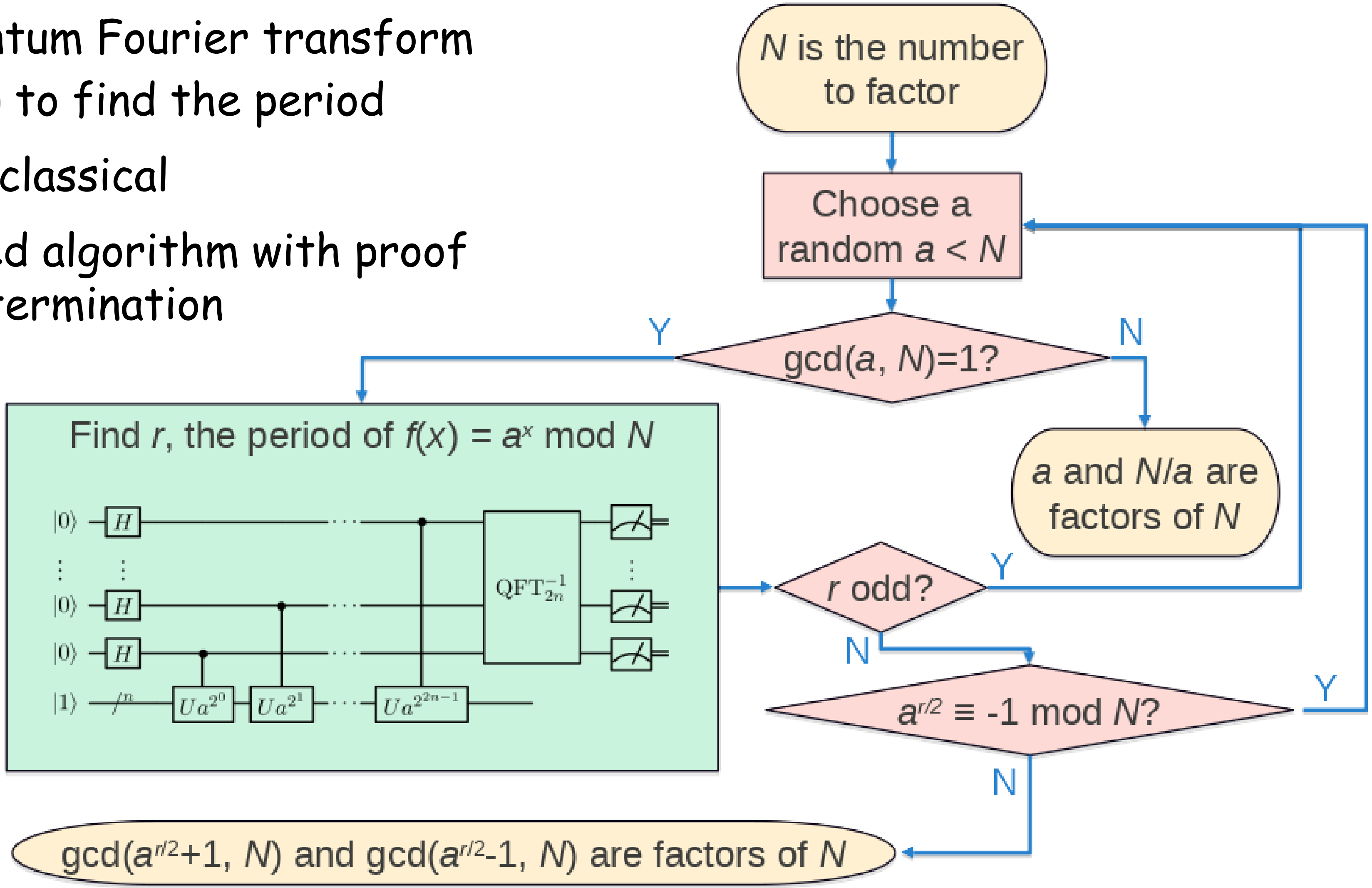
- Then in a May 2014 book titled, "Fast quantum modular exponentiation architecture for Shor's factoring algorithm" (pp0649-0682) by Archimedes Pavlidis and Dimitris Gizopoulos, the authors proposed a 9n+2 implementation, adding a significant number of qubits to reduce circuit depth.
- Breaking 2,048-bit RSA encryption would require **18,434 logical qubits** or roughly **18 million physical qubits**. One reason to minimize circuit depth is that qubits lose "coherence" over time.
- The longer it takes to execute a circuit, as determined, in part, by circuit depth, the more noise can seep into the system and the more errors can pop up in the results. Even today, one way to reduce errors is to **minimize circuit depth.**

- Most recently, in an April 13, 2021 paper titled, “How to factor 2048 bit RSA integers in **8 hours** using 20 million noisy qubits” by Craig Gidney and Martin Eker, the authors estimate that breaking RSA encryption, specifically, would require roughly **20,000 logical qubits**, or approximately 20 million physical qubits.
- The authors take one step further and quantify the runtime for their configuration as only eight hours.
- Compare that to the estimate for the world’s most powerful supercomputers to break RSA encryption, which is well beyond any human lifetime, and the power of Shor’s Factoring Algorithm becomes clear.

- Again, Shor’s algorithm runtime is **inversely** proportional to the number of logical qubits available. It also depends on the **qubit technology** in use, since different quantum computers execute operations at vastly different speeds.
- The **fewer** qubits needed, the **longer** the algorithm will take to run, but the sooner modern cryptosystems transition from obsolescent to obsolete.
- Conversely, the more qubits available, the faster modern cryptosystems will be cracked. But fortunately, those days are even further away.

Shor's Algorithm

- Use a quantum Fourier transform
  - (QFT) to find the period
- All else is classical
- Randomized algorithm with proof of timely termination



N is the number to factor

Choose a random *a* < *N*

gcd(*a*, *N*)=1?

Y

N

a and *N*/*a* are factors of *N*

Find *r*, the period of *f*(*x*) = a^x mod *N*

r odd?

Y

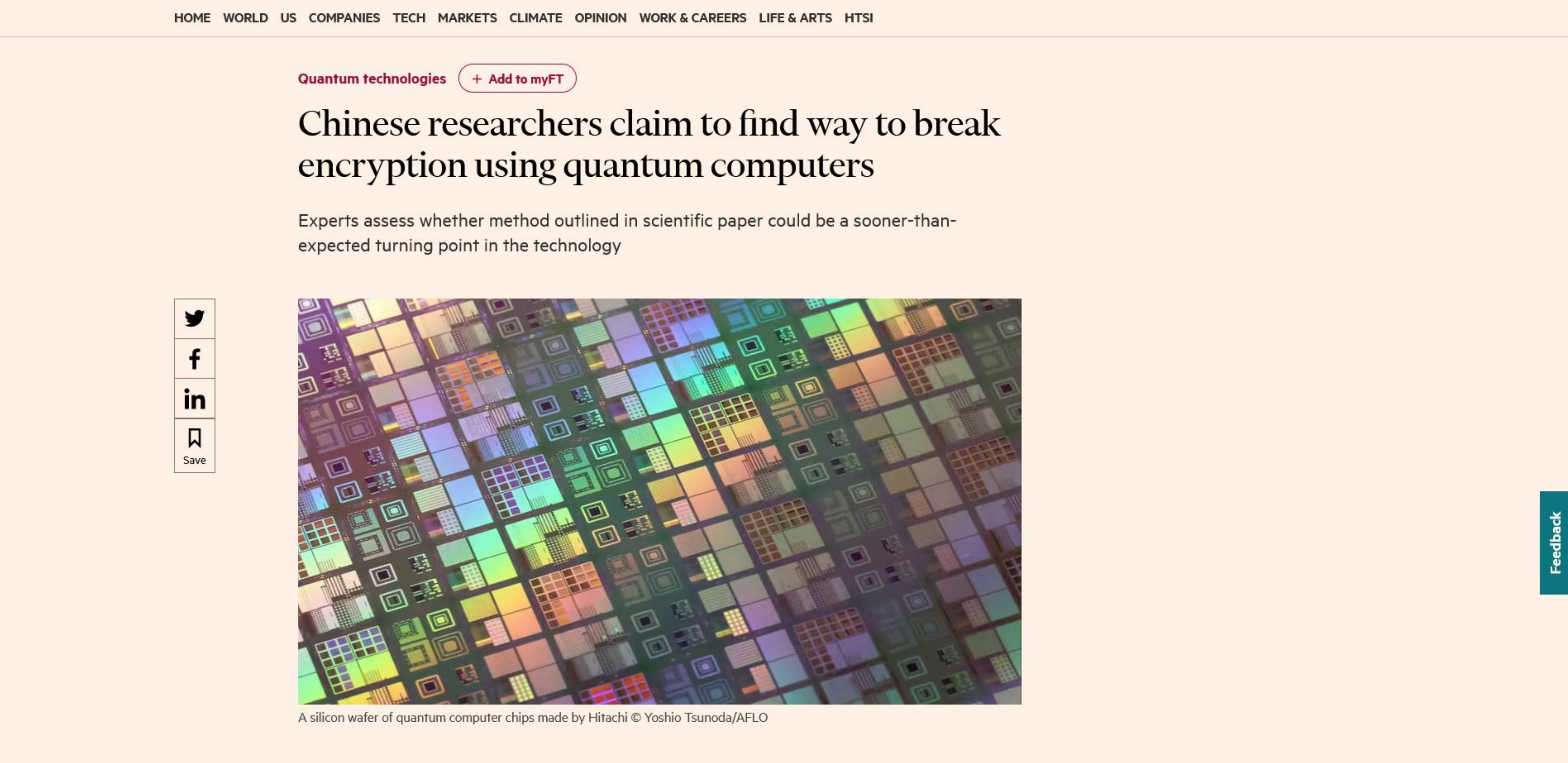
N

$a^{r/2} \equiv -1 \bmod N$?

Y

N

gcd($a^{r/2}$+1, *N*) and gcd($a^{r/2}$-1, *N*) are factors of *N*


HOME WORLD US COMPANIES TECH MARKETS CLIMATE OPINION WORK & CAREERS LIFE & ARTS HTSI
+ Add to myFT
Save
Feedback

Quantum technologies

Chinese researchers claim to find way to break encryption using quantum computers

Experts assess whether method outlined in scientific paper could be a sooner-than-expected turning point in the technology

A silicon wafer of quantum computer chips made by Hitachi © Yoshio Tsunoda/AFLO

Chinese researchers claim to find way to break encryption using quantum computers

- Experts assess whether method outlined in scientific paper could be a sooner-than-expected turning point in the technology
- Computer security experts were struggling this week to assess a startling claim by Chinese researchers that they have found a way to break the most common form of online encryption using the current generation of quantum computers, years before the technology was expected to pose a threat.

- The method, outlined in a scientific paper published in late December 2022, could be used to break the RSA algorithm that underpins most online encryption using a quantum machine with only **372** qubits — or quantum bits, a basic unit of quantum computing — according to the claims from 24 researchers from a number of academic bodies and state laboratories.

- IBM has already said that its **433 qubit Osprey system**, the most powerful quantum computer to have been publicly unveiled, will be made available to its customers early this year.
- If correct, the research would mark a significant moment in the history of computer security, said Roger Grimes, a computer security expert and author. “It’s a huge claim,” he said. “It would mean that governments could crack other governments secrets.

- If it’s true — a big if — it would be a secret like out of the movies, and one of the biggest things ever in computer science.” Other experts said that while the theory outlined in the research paper appeared sound, trying to apply it in practice could well be beyond the reach of today’s quantum technology.

- “As far as I can tell, the paper isn’t wrong,” said **Peter Shor**, the Massachusetts Institute of Technology scientist whose 1994 algorithm proving that a quantum machine could defeat online encryption helped to trigger a research boom in quantum computing. Shor’s method requires machines with many hundreds of thousands, or even millions, of qubits, something that many experts believe is a decade or more away.

- Shor added, however, that the Chinese researchers had “failed to **address how fast the algorithm will run**”, and said that it was possible it “will still take millions of years”.
- He said: “In the absence of any analysis showing that it will be faster, I suspect that the most likely scenario is that it’s not much of an improvement.”

- The latest research paper is the second time in less than a year that the field of computer security has been jolted by claims that online encryption was in imminent danger of being broken.
- German mathematician Claus-Peter Schnorr published an algorithm last year that he said was a far more efficient way to factor large prime numbers — central to breaking the RSA code — potentially putting it within reach of traditional, or “classical” computers.
- But it turned out that Schnorr’s technique could not be scaled up to work as needed to challenge the RSA algorithm.

- The latest research paper claims to make up for the gap in Schnorr’s research by using a quantum computer to speed up the part of the calculation he was unable to solve.
- It highlights the use of hybrid techniques that combine quantum and classical systems, the current focus of much of the work that is going on to find practical uses for quantum machines.

- The Chinese researchers said they had used their algorithm to factor a number with 48 bits on a quantum computer with 10 qubits, but that they had not had the chance to try to scale it up to work on a much bigger system.

- Computer security expert Bruce Schneier said that the paper had left open the question **of whether the technique would work in practice**. “We have no empirical proof that the [new] quantum algorithm overcomes the Schnorr scaling problem,” he said. “There’s no reason to believe it won’t — but there’s no reason to believe it will.”
- He added that quantum systems had already reached the scale outlined by the researchers, meaning that their claims could be put to the test very soon.

- Even if the research claim proved unfounded, Schneier said it highlights a race to find a way to break encryption using quantum computers far earlier than many had expected. “The betting is, as in all these cases, breaking RSA won’t work.
- But some day that bet will be wrong.”

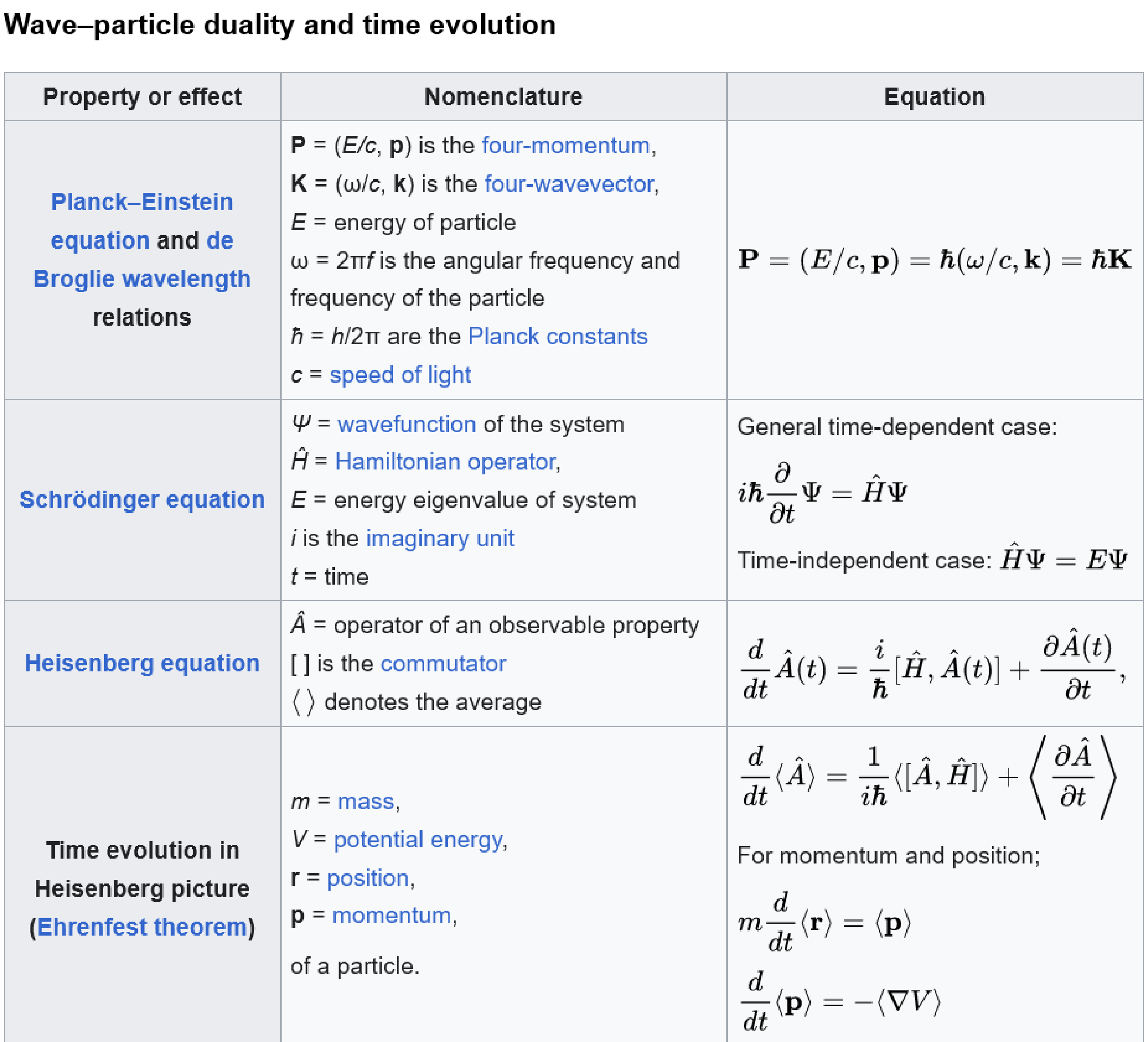
Important Quantum Computing Equations

- Schrödinger equation
- Heisenberg uncertainty principle
- Dirac notation

- Bra-ket notation
- Born rule
- Liouville equation
- Master equation

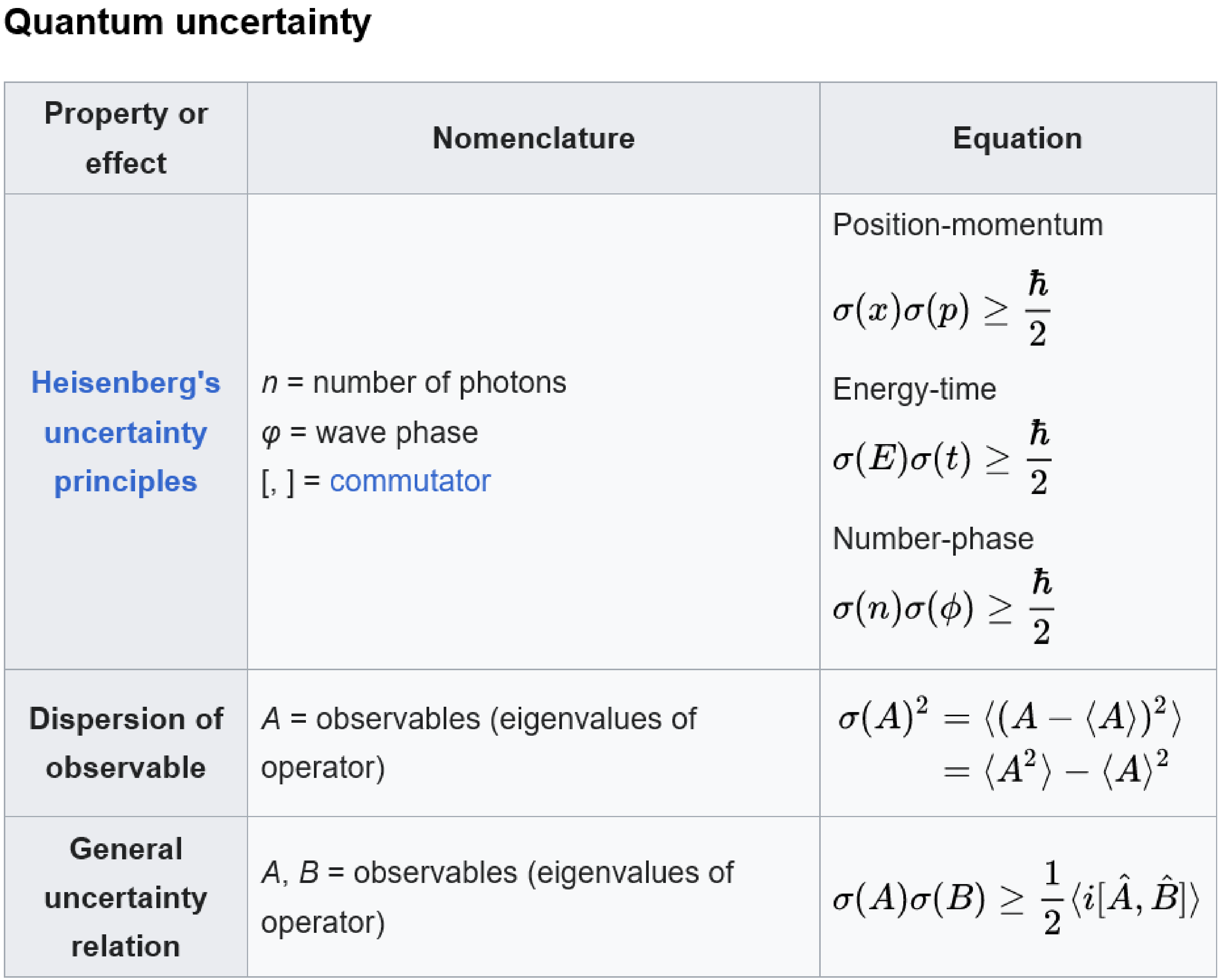
- Hubbard model
- Quantum Fourier transform
- Quantum gates (Hadamard, NOT, CNOT, etc.)
- Tensor product

- Projection operator
- Density matrix
- Bloch sphere representation.



Wave–particle duality and time evolution

Property or effect	Nomenclature	Equation
Planck–Einstein equation and de Broglie wavelength relations	**P** = (*E*/*c*, **p**) is the four-momentum, **K** = (ω/*c*, **k**) is the four-wavevector, *E* = energy of particle ω = 2π*f* is the angular frequency and frequency of the particle ħ = *h*/2π are the Planck constants *c* = speed of light	$\mathbf{P} = (E/c, \mathbf{p}) = \hbar(\omega/c, \mathbf{k}) = \hbar\mathbf{K}$
Schrödinger equation	*Ψ* = wavefunction of the system *Ĥ* = Hamiltonian operator, *E* = energy eigenvalue of system *i* is the imaginary unit *t* = time	General time-dependent case: $i\hbar\frac{\partial}{\partial t}\Psi = \hat{H}\Psi$ Time-independent case: $\hat{H}\Psi = E\Psi$
Heisenberg equation	*Â* = operator of an observable property [ ] is the commutator ⟨ ⟩ denotes the average	$\frac{d}{dt}\hat{A}(t) = \frac{i}{\hbar}[\hat{H}, \hat{A}(t)] + \frac{\partial \hat{A}(t)}{\partial t},$
Time evolution in Heisenberg picture (Ehrenfest theorem)	*m* = mass, *V* = potential energy, **r** = position, **p** = momentum, of a particle.	$\frac{d}{dt}\langle\hat{A}\rangle = \frac{1}{i\hbar}\langle[\hat{A}, \hat{H}]\rangle + \left\langle\frac{\partial \hat{A}}{\partial t}\right\rangle$ For momentum and position; $m\frac{d}{dt}\langle\mathbf{r}\rangle = \langle\mathbf{p}\rangle$ $\frac{d}{dt}\langle\mathbf{p}\rangle = -\langle\nabla V\rangle$

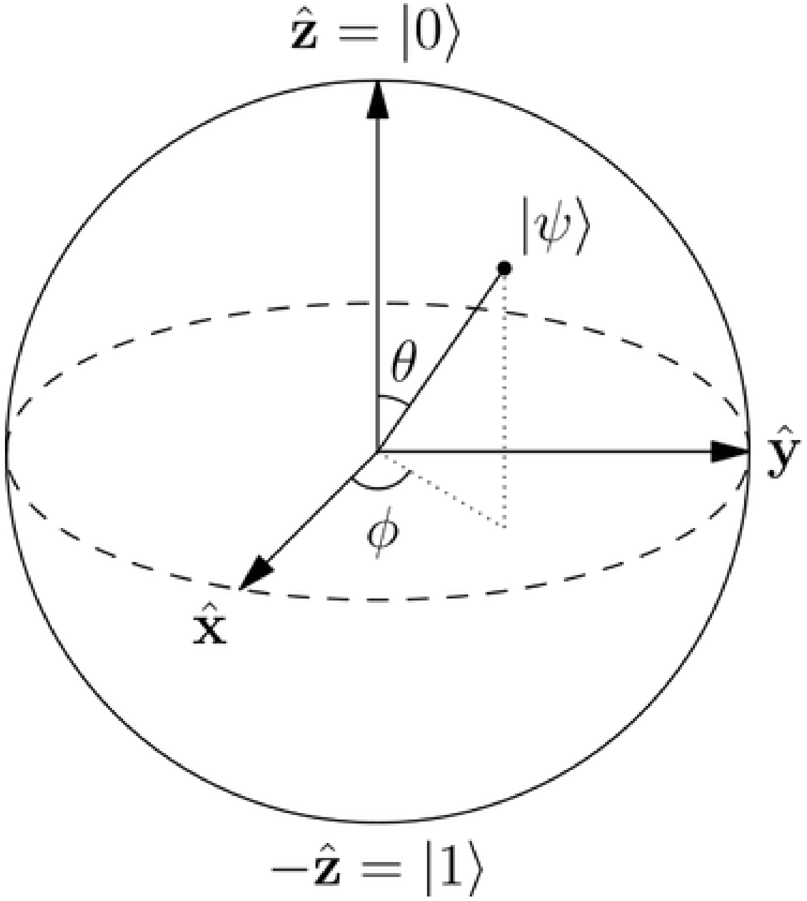


Quantum uncertainty

Property or effect	Nomenclature	Equation
Heisenberg's uncertainty principles	*n* = number of photons *φ* = wave phase [,] = commutator	Position-momentum $\sigma(x)\sigma(p) \geq \frac{\hbar}{2}$ Energy-time $\sigma(E)\sigma(t) \geq \frac{\hbar}{2}$ Number-phase $\sigma(n)\sigma(\phi) \geq \frac{\hbar}{2}$
Dispersion of observable	*A* = observables (eigenvalues of operator)	$\begin{aligned}\sigma(A)^2 &= \langle (A - \langle A \rangle)^2 \rangle \\ &= \langle A^2 \rangle - \langle A \rangle^2\end{aligned}$
General uncertainty relation	*A*, *B* = observables (eigenvalues of operator)	$\sigma(A)\sigma(B) \geq \frac{1}{2}\langle i[\hat{A}, \hat{B}] \rangle$

The Bloch sphere representation

The above result into the *Bloch sphere representation*, named after Felix Bloch. This representaiton is a unit 2-sphere, where at the north and south poles lie two mutually This representation has a nice geometric perspective:



Quantum Computation

Classical Computation:

Classical logic bit: “0” and “1”

Quantum Computation:

Quantum bit, “Qubit”, can be manipulated using the rules of quantum physics

Orthogonal quantum states |0> , |1> and their superposition $|\Psi> = c_0|0> + c_1|1>$

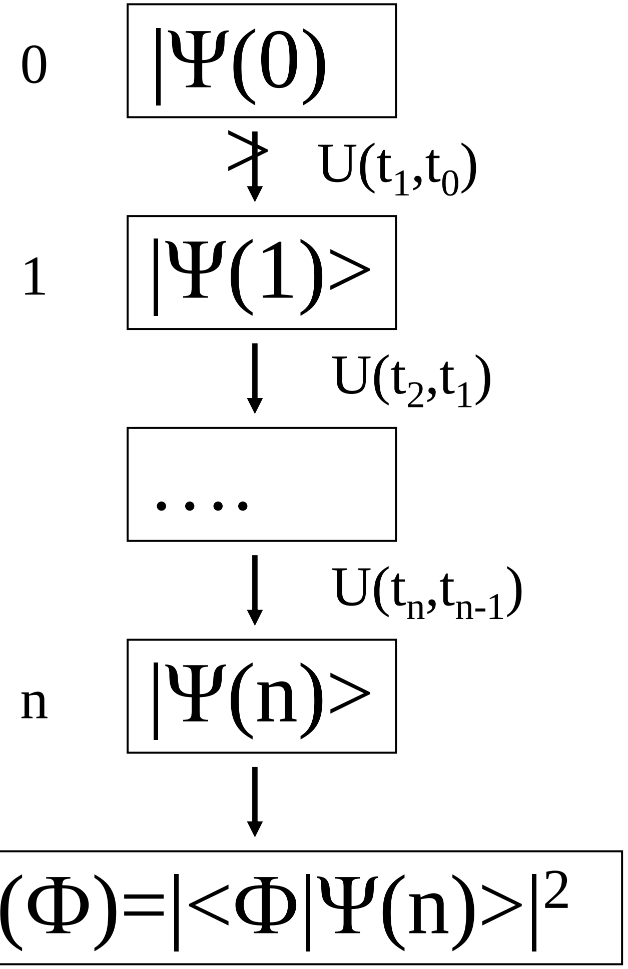
A Quantum state of M bits is a superposition of 2^M states.

The quantum computation is a parallel computation in which all 2^M basis vectors are acted upon at the same time.

If one wanted to simulate a quantum computer using a classical computer one would need to multiply together 2^M dimensional unitary matrices, to simulate each step.

A quantum computer can factorize a 250-digit number in seconds while an ordinary computer will take 800 000 years!

Quantum Computation



0 $|\Psi(0)$

$U(t_1,t_0)$

1 $|\Psi(1)>$

$U(t_2,t_1)$

….

$U(t_n,t_{n-1})$

n $|\Psi(n)>$

$P(\Phi)=|<\Phi|\Psi(n)>|^2$

Preparation:

The initial preparation of the state defines a wave function at time $t_0=0$.

State evolution:

Evolved by a sequence of unitary operations

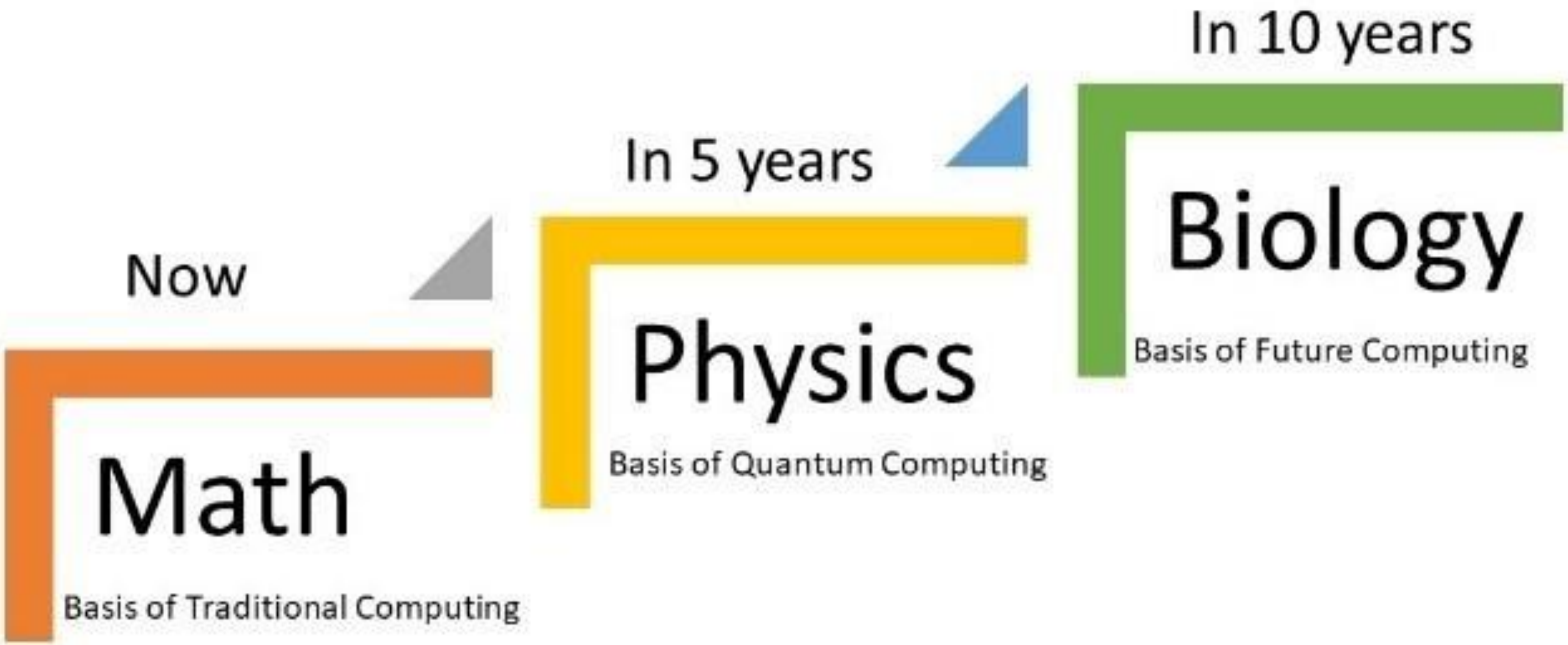
Measurement:

Quantum measurement is projective.

Collapsed by measurement of the state

Quantum Computing Categories

- Quantum Computing is the area of study focused on developing computer technology based on the principles of quantum theory.
- Tens of billions of public and private capitals are being invested in Quantum technologies.
- Countries across the world have realized that quantum technologies can be a major disruptor of existing businesses, they have collectively invested $24 billion in in quantum research and applications in 2021



Now

Math

Basis of Traditional Computing

In 5 years

Physics

Basis of Quantum Computing

In 10 years

Biology

Basis of Future Computing

Future of Computing

- **Physical vs Logical Qubits**
- When discussing quantum computers with error correction, we talk about physical and logical qubits.
- Physical qubits are the physical qubits in quantum computer, whereas logical qubits are groups of physical qubits we use as a single qubit in our computation to fight noise and improve error correction.

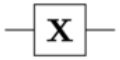
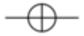
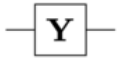
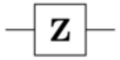
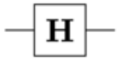
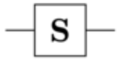
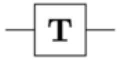
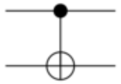
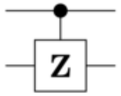
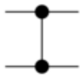
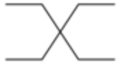
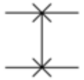
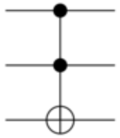
- To illustrate this, let’s consider an example of a quantum computer with 100 qubits. Let’s say this computer is prone to noise, to remedy this we can use multiple qubits to form a single more stable qubit.
- We might decide that we need 10 physical qubits to form one acceptable logical qubit.
- In this case we would say our quantum computer has 100 physical qubits which we use as 10 logical qubits.

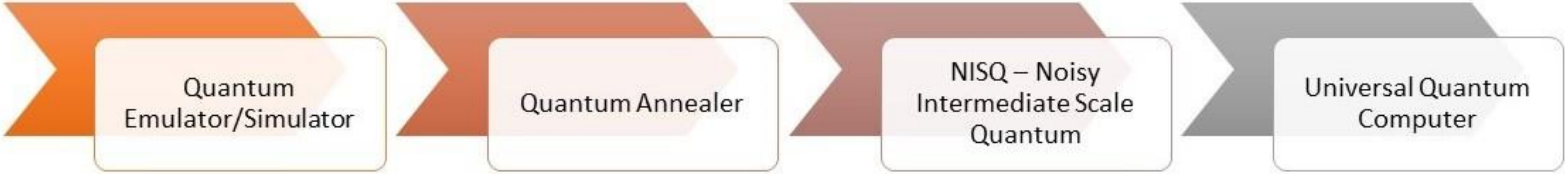
- Distinguishing between physical and logical qubits is important. There are many estimates as to how many qubits we will need to perform certain calculations, but some of these estimates talk about logical qubits and others talk about physical qubits.
- For example: To break RSA cryptography we would need thousands of logical qubits but millions of physical qubits.

- Another thing to keep in mind, in a classical computer compute-power increases **linearly** with the number of transistors and clock speed, while in a Quantum computer compute-power increases **exponentially** with the addition of each logical qubit

Quantum Gate

- In quantum computing and specifically the quantum circuit model of computation, a **quantum logic gate** (or simply **quantum gate**) is a basic quantum circuit operating on a small number of qubits. They are the building blocks of quantum circuits, like classical logic gates are for conventional digital circuits.

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)			$\begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$
π/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)			$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$



Quantum Emulator/Simulator

Quantum Annealer

NISQ – Noisy Intermediate Scale Quantum

Universal Quantum Computer

Quantum Computers Categories

- **Quantum computers fall into four categories**

1. Quantum Emulator/Simulator
2. Quantum Annealer
3. Noisy Intermediate Scale Quantum (NISQ)
4. Universal Quantum Computer – which can be a Cryptographically Relevant Quantum Computer (CRQC)

- **Quantum Emulator/Simulator**
- These are **classical computers** that you can buy today that simulate quantum algorithms. They make it easy to test and debug a quantum algorithm that someday may be able to run on a Universal Quantum Computer (UQC).
- Since they don’t use any quantum hardware, they are no faster than standard computers.

- **Quantum Annealer**
- A special purpose quantum computer designed to only **run combinatorial optimization problems**, not general-purpose computing, or cryptography problems.
- While they have more physical Qubits than any other current system they are not organized as gate-based logical qubits.
- Currently this is a commercial technology in search of a future viable market.

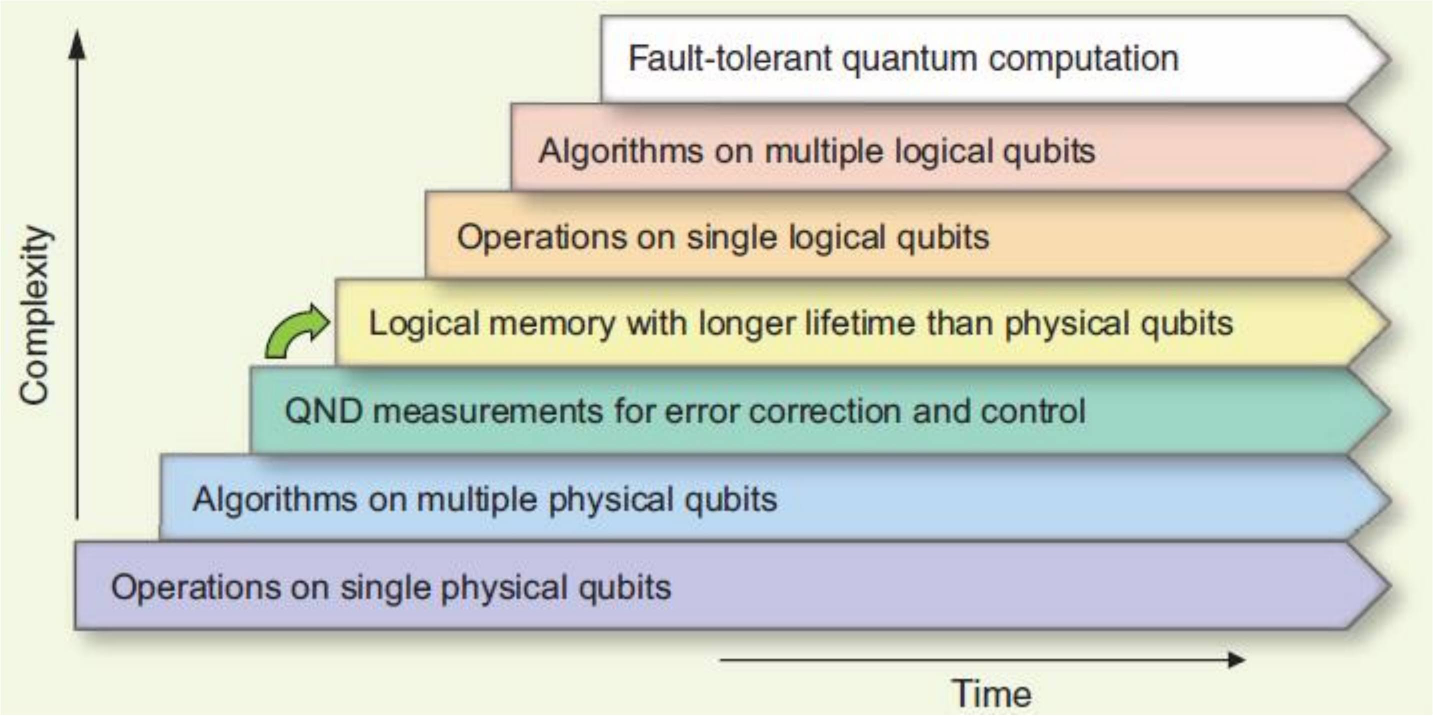
- **Noisy Intermediate-Scale Quantum (NISQ) computers.**
- Think of these as *prototypes* of a Universal Quantum Computer – with several orders of magnitude fewer bits.
- They currently have 50-100 qubits, limited gate depths, and short coherence times.

- As they are short several orders of magnitude of Qubits, NISQ computers cannot perform any useful computation, however they are a necessary phase in the learning, especially to drive total system and software learning in parallel to the hardware development.
- Think of them as the **training wheels** for future universal quantum computers.

- **Universal Quantum Computers / Cryptographically Relevant Quantum Computers (CRQC**)
- This is the **ultimate goal**. If you could build a universal quantum computer with fault tolerance (i.e., millions of error- corrected physical qubits resulting in thousands of logical Qubits), you could run quantum algorithms in cryptography, search and optimization, quantum systems simulations, and linear equations solvers.

Steps to Build a Quantum Computer

- Roadmap toward realizing a quantum computer


Fault-tolerant quantum computation
Algorithms on multiple logical qubits
Operations on single logical qubits
Logical memory with longer lifetime than physical qubits
QND measurements for error correction and control
Algorithms on multiple physical qubits
Operations on single physical qubits
Complexity
Time

M. H. Devoret and R. J. Schoelkopf, *Science*, **339**, 1169 (2013).