Quantum Computing

(in way too short a time)

Riccardo Pucella

April 2021

♫ It's a small world... ♫

Scott Aaronson Receives 2020 ACM Prize in Computing

April 14, 2021

ACM has named Scott Aaronson of the University of Texas at Austin the recipient of the 2020 ACM Prize in Computing for groundbreaking contributions to quantum computing.

Aaronson showed how results from computational complexity theory can provide new insights into the laws of quantum physics, and brought clarity to what quantum computers will, and will not, be able to do. He helped develop the concept of quantum supremacy, which denotes the milestone that is achieved when a quantum device can solve a problem that no classical computer can solve in a reasonable amount of time. His quantum supremacy experiments allow scientists to give convincing evidence that quantum computers provide exponential speedups, without having to first build a full fault-tolerant quantum computer.

Aaronson is the David J. Bruton Jr. Centennial Professor of Computer Science at the University of Texas at Austin.

In the beginning...

David Deutsch, **Quantum theory, the Church–Turing principle and the universal quantum computer** (1985)

Investigates a physical-system version of the Church-Turing thesis:
Every finitely realizable physical system can be perfectly simulated by a universal model computing machine operating by finite means'.

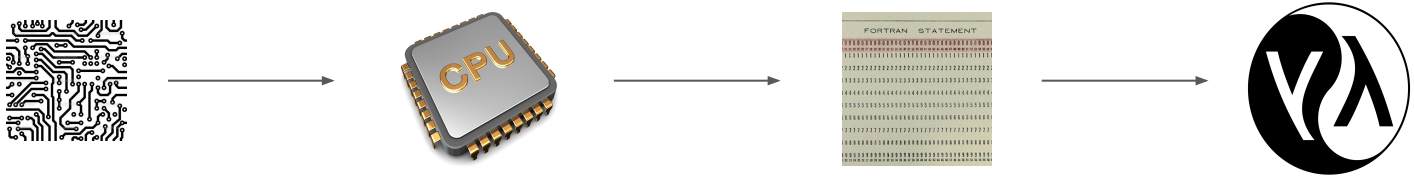
He proposes a computational model based on quantum physics ("quantum computing") as such a universal computing machine

From proposal to … reality?

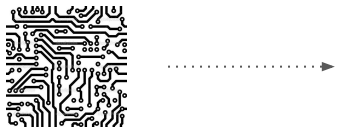
Classical computation

From proposal to … reality?

Classical computation



Quantum computation



(1)

Quantum bits

Bits

The basic element of information in classical computation is the **bit**

A bit is a "thing" that has two states that can be distinguished

- 0 or 1
- ON or OFF
- either of two configurations of electric charges on metal plates

Bits can be physically realized in many different ways

- talk to a computer engineer

For computations, we don't care how bits are realized, just that they can be

- and that you can physically realize operations on such bits

Quantum bits

In quantum computing, the basic element of information is the **quantum bit (qbit)**

A qbit is a "thing" that has a state

- state of a qbit is a vector in $\mathbb{C}^2$

$\mathbb{C}^2$ has a basis — two vectors from which you can derive all others by linearity

- Write them $|0\rangle$ and $|1\rangle$
- Notation suggest they will play the role of classical 0 and classical 1

The state of a qbit is a vector $a|0\rangle + b|1\rangle$ e.g., $0.6|0\rangle + 0.8|1\rangle$

Can be represented by a column vector: $\begin{pmatrix} a \\ b \end{pmatrix}$

We normalize states to be unit vectors (length 1) for technical reasons

Quantum bits

In quantum computing, the basic element of information is the **quantum bit (qbit)**

A qbit is a "thing" that has a state

- state of a qbit is a vector in $\mathbb{C}^2$

$\mathbb{C}^2$

-

-

The quantum state of a qubit is a vector of unit length in a two-dimensional complex vector space

The state of a qbit is a vector $a|0\rangle + b|1\rangle$ e.g., $0.6|0\rangle + 0.8|1\rangle$

Can be represented by a column vector: $\begin{pmatrix} a \\ b \end{pmatrix}$

We normalize states to be unit vectors (length 1) for technical reasons

Superposed states

A state such as 0.6|0〉 + 0.8|1〉 is a **superposition** of |0〉 and |1〉

- think of it as being in a state that's both |0〉 and |1〉 at the same time but maybe a bit more |1〉 than it is |0〉?

That's the key to why quantum computation is interesting: the state of a qbit can be both |0〉 and |1〉 at the same time

It gives us a lot more possible intermediate results that we can use to encode useful information during the computation

- not just |0〉 and |1〉, but combinations of |0〉 and |1〉
- how we use those intermediate results is the "art" of quantum programming

Wait… but what's a qbit really?

Well, what's a bit? It's a concept. It can be realized physically. Somehow.

You can realize qbits with chilled superconductors wires, laser-trapped ions, photons. ... (Or so they tell me.)

1. What do $|0\rangle$ and $|1\rangle$ mean?
   Depends on the implementation — they generally correspond to observable physical states (trajectory of a photon, say)

2. Why is a state a vector in $\mathbb{C}^2$?
   Excellent question — it took 25 years in the early 20th century and the greatest physicists of the time to develop a quantum theory that explains this

(2)

Quantum logic gates

Classical logic gates

In the classical world, logic gates represent the operations you can perform on bits — basically, they're functions from Boolean values to Boolean values.

They can be physically realized, and how they are realized depends on how bits are realized

Classical computation = bits + Boolean logic

Quantum computation = qbits + ??

The NOT quantum gate

What would a NOT gate look like in the quantum world?

If $|0\rangle$ and $|1\rangle$ play the role of 0 and 1, then:

$$\text{NOT } |0\rangle = |1\rangle$$
$$\text{NOT } |1\rangle = |0\rangle$$

What if the state is superposed?

We extend linearly:

$$\text{NOT } a|0\rangle + b|1\rangle = a \text{ NOT } |0\rangle + b \text{ NOT } |1\rangle$$
$$= a|1\rangle + b|0\rangle$$

Matrix representation of the NOT gate

NOT is linear operator that transforms a vector in $\mathbb{C}^2$ into a vector in $\mathbb{C}^2$:

$$\text{NOT} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} b \\ a \end{pmatrix}$$

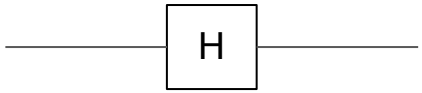
We know we can represent such a transformation by a matrix:

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \longrightarrow \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} b \\ a \end{pmatrix}$$

The Hadamard gate

$H|0\rangle = (|0\rangle + |1\rangle) / \sqrt{2}$

$H|1\rangle = (|0\rangle - |1\rangle) / \sqrt{2}$



Again, extends by linearity to superposed states:

$H\, a|0\rangle + b|1\rangle = a(|0\rangle + |1\rangle) / \sqrt{2} + b(|0\rangle - |1\rangle) / \sqrt{2}$

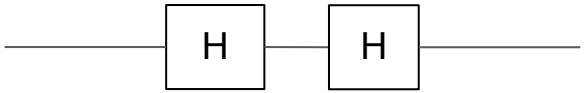
$= (a + b) / \sqrt{2}\, |0\rangle + (a - b) / \sqrt{2}\, |1\rangle$

As a matrix:

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

The Hadamard gate

Check: HH = $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$



So H takes a state such as |0⟩ and "mixes it up" but you can still recover |0⟩

The rough way many such algorithms work is to first use Hadamard gates to "spread out" in quantum states like |0⟩ + |1⟩ (or many-qubit analogs), i.e., in superpositions of multiple 2 computational basis states.

At the end of the algorithm they use clever patterns of cancellation and reinforcement to bring things back together again into one (or possibly a few, in the many-qubit case) computational basis state, containing the desired answer.

Exercise left to the reader

What does this do to a qbit?

$a|0\rangle + b|1\rangle \quad \rightarrow \quad a|0\rangle - b|1\rangle$

General (single qbit) quantum logic gates

A quantum logic gate is any unitary matrix over $\mathbb{C}^2$

- Unitary : takes unit vectors in $\mathbb{C}^2$ to unit vectors in $\mathbb{C}^2$
- multiplying it by its conjugate transpose gives the identity

E.g.:

Y = $\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$ Z = $\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$ Rotation by θ = $\begin{pmatrix} cos\,\theta & -sin\,\theta \\ sin\,\theta & cos\,\theta \end{pmatrix}$

Any unitary matrix can be a quantum logic gate

We usually restrict to a finite set of gates that are *universal*

(3)

Observations

Measurement "in the computational basis"

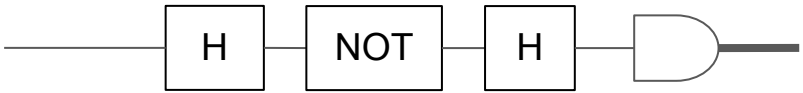
In the real world, you cannot observe (measure) a superposed state. When you measure a qbit, it collapses to $|0\rangle$ or $|1\rangle$. Welcome to quantum mechanics.

Measurement "in the computational basis":

- If a qbit is in state $a|0\rangle + b|1\rangle$ and you measure it, you get $|0\rangle$ with probability $|a|^2$ and $|1\rangle$ with probability $|b|^2$

The outcome of measurement is classical information: $|0\rangle$ or $|1\rangle$

Measurement is the last step of a quantum computation



Example

Say that in the midst of a computation you produce one of

$(|0\rangle + |1\rangle) / \sqrt{2}$ or $(|0\rangle - |1\rangle) / \sqrt{2}$

If you measure them, you cannot distinguish them — you get $|0\rangle$ and $|1\rangle$ both with probability 0.5.

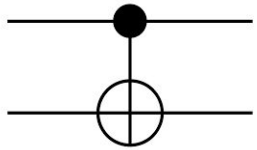
But if we run them through an Hadamard gate and then measure, we get $|0\rangle$ with probability 1 in one case and $|1\rangle$ with probability 1 in the other case.

(4)

Multi-qbit gates

Controlled-NOT gate

Here's an example of a 2-qbit logic gate:



- if control qbit is $|1\rangle$, the target qbit is flipped
- if control qbit is $|0\rangle$, the target qbit is unflipped.

What is the state of a 2-qbit system? It's a pair of states of each qbit

- it's a vector in $\mathbb{C}^2 \times \mathbb{C}^2 = \mathbb{C}^4$
- basis of $\mathbb{C}^4$ = B x B where B is a basis of $\mathbb{C}^2$
- so a basis of $\mathbb{C}^4$ is $|00\rangle$, $|01\rangle$, $|10\rangle$, and $|11\rangle$

If q_1 is $a|0\rangle + b|1\rangle$ and q_2 is $c|0\rangle + d|1\rangle$, then

$(q_1, q_2) = q_1 q_2 = ac|00\rangle + ad|01\rangle + bc|10\rangle + bd|11\rangle$ (a unit vector in $\mathbb{C}^4$)

Controlled-NOT gate

CNOT $|00\rangle = |00\rangle$
CNOT $|01\rangle = |01\rangle$
CNOT $|10\rangle = |11\rangle$
CNOT $|11\rangle = |10\rangle$

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

CNOT $|x,y\rangle = |x, y \oplus x\rangle$

Extend to superposed states by linearity

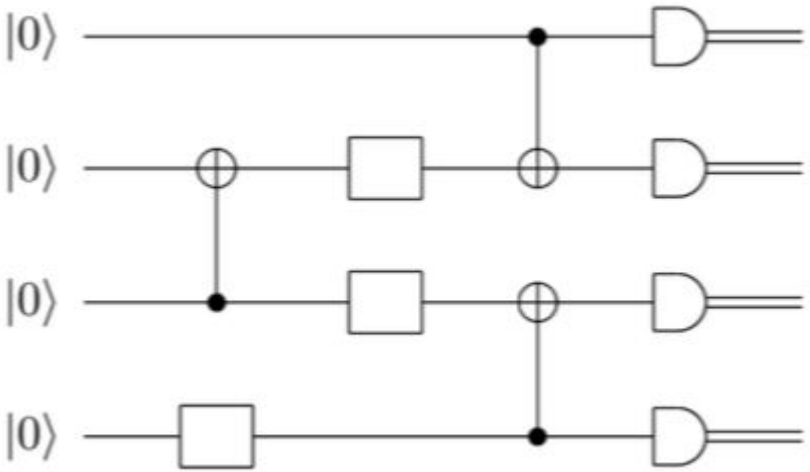
CNOT $a|00\rangle + b|01\rangle + c|10\rangle + d|11\rangle = a|00\rangle + b|01\rangle + d|10\rangle + c|01\rangle$

Things get interesting when the control qbit, say, is in a superposed state:

CNOT $(|0\rangle + |1\rangle / \sqrt{2})(|0\rangle - |1\rangle / \sqrt{2}) = (|0\rangle - |1\rangle / \sqrt{2})(|0\rangle - |1\rangle / \sqrt{2})$

The control bit changes, the target bit remains the same

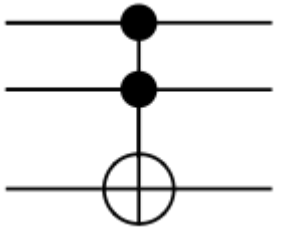
Generalize to circuits with more qbits



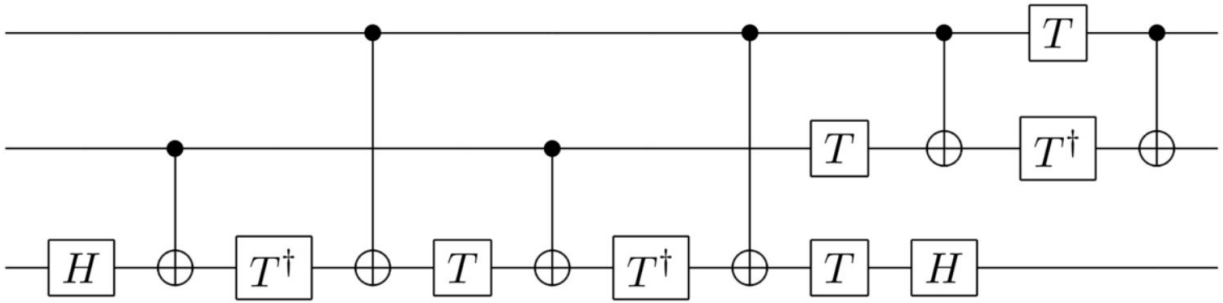
The Toffoli gate

Think of it as a controlled AND

- if both control qbits are |1⟩ the target qbit is flipped
- if either control qbit is |0⟩, the target qbit is not flipped



Fun fact:



$$\text{where } T = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}.$$

So why do we care?

Quantum computation doesn't let you compute more things — it lets you compute some things faster by taking advantage of state superposition

You go from classical states (inputs) to classical states (outputs after measuring) but taking "shortcuts" through superposed states