Quantum Technology at CERN

Leonardo Lavagna

THE BIG STORY

The Quantum Apocalypse Is Coming. Be Very Afraid

What happens when quantum computers can finally crack encryption and break into the world's best-kept secrets? It's called Q-Day—the worst holiday maybe ever.

AMIT KATWALA

MAR 24, 2025 6:00 AM

Meet Willow, our state-of-the-art quantum chip

Dec 09, 2024
9 min read

Our new chip demonstrates error correction and performance that paves the way to a useful, large-scale quantum computer

Last updated: June 12, 2025

Today I'm delighted to announce Willow, our latest quantum chip. Willow has state-of-the-art performance across a number of metrics, enabling two major achievements.

- The first is that Willow can reduce errors exponentially as we scale up using *more* qubits. This cracks a key challenge in quantum error correction that the field has pursued for almost 30 years.
- Second, Willow performed a standard benchmark computation in under five minutes that would take one of today's fastest supercomputers 10 septillion (that is, 10^{25}) years — a number that vastly exceeds the age of the Universe.

called G-Day—the worst holiday maybe ever.

AMIT KATWALA

MAR 24, 2025 6:00 AM

Meet Willow, quantum chi

Dec 09, 2024
9 min read

Our new chip demonstrate
quantum computer

Last updated: June 12, 2025

Today I'm delighted to announ
performance across a number

- The first is that Willow can
cracks a key challenge in q
- Second, Willow performed
take one of today's fastest
vastly exceeds the age of t

Google unveils 'mindboggling' quantum computing chip

Chip takes minutes to complete tasks that would otherwise take 10,000,000,000,000,000,000,000,000 years

Dr Erik Lucero, lead engineer of Google Quantum AI, points to motherboards while leading media on a tour of the Quantum Computing Lab in September 2022. Photograph: Frederic J Brown/AFP/Getty Images

NEWSLETTERS SI

Meet Willow, quantum chip

Dec 09, 2024
9 min read

Last up

Today I'
perform

- The
  crac
- Seco
  take
  vast

Google unveils 'mindboggling' quantum computing chip

NEWSLETTERS

SECTIONS

NEW YORK POST

Science Space Environment Wildlife Archaeology

DEAL DROP 101+ best Memorial Day 2026 sales and deals we're tracking so far

SCIENCE

Google claims quantum chip may prove existence of parallel universes

By Ben Cost

Published Dec. 15, 2024, 4:58 p.m. ET

media on a tour of the Quantum Computing Lab in September 2022. Photograph: Frederic J Brown/AFP/Getty Images


Human Genome Project
Article
Talk
From Wikipedia, the free encyclopedia


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


"THE TALK"
BY SCOTT AARONSON & ZACH WEINERSMITH
WHAT ARE YOU READING?!
Popular Scientism
Quantum
NOTHING! NOTHING!
YOU'RE GROWING UP SO FAST. I... I THINK IT'S TIME YOU AND I HAD... "THE TALK"
THE QUANTUM COMPUTING TALK.

- Overview
- Interactive Discussion on Quantum Computing and Algorithms

Noisy intermediate-scale quantum computing

From Wikipedia, the free encyclopedia

(Redirected from Noisy intermediate-scale quantum era)

Noisy intermediate-scale quantum (**NISQ**) **computing** is characterized by quantum processors containing up to 1,000 qubits which are not advanced enough yet for fault-tolerance or large enough to achieve quantum advantage.[1][2] These processors, which are sensitive to their environment (noisy) and prone to quantum decoherence, are not yet capable of continuous quantum error correction. This intermediate-scale is defined by the quantum volume, which is based on a moderate number of qubits and gate fidelity. The **NISQ era** is the current state of quantum computer technology,[3][4][5] and the term was coined by John Preskill in 2018.[6][4]

According to Microsoft Azure Quantum's scheme, NISQ computation is considered level 1, the lowest of the quantum computing implementation levels.[7][8]

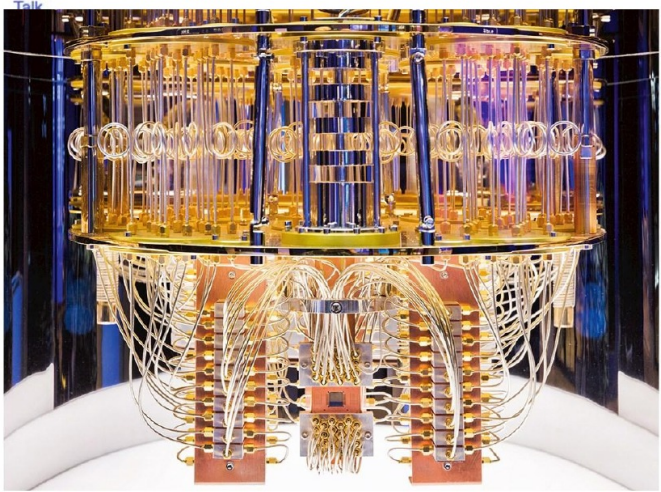
In October 2023, the 1,000 qubit mark was passed for the first time by Atom Computing's 1,180 qubit quantum processor.[9] However, as of 2024, only two quantum processors have over 1,000 qubits, with sub-1,000 quantum processors still remaining the norm.[10]

Noisy intermediate-scale quantum computing

7 languages

Article Talk

Read Edit View history Tools



; containing up to 1,000 qubits which
1][2] These processors, which are
continuous quantum error correction.
r of qubits and gate fidelity. The
r John Preskill in 2018.[6][4]

west of the quantum computing

ubit quantum processor.[9] However,
ssors still remaining the norm.[10]

Noisy intermediate-scale quantum computing

s containing up to 1,000 qubits which
1][2] These processors, which are
continuous quantum error correction.
r of qubits and gate fidelity. The
John Preskill in 2018.[6][4]

west of the quantum computing

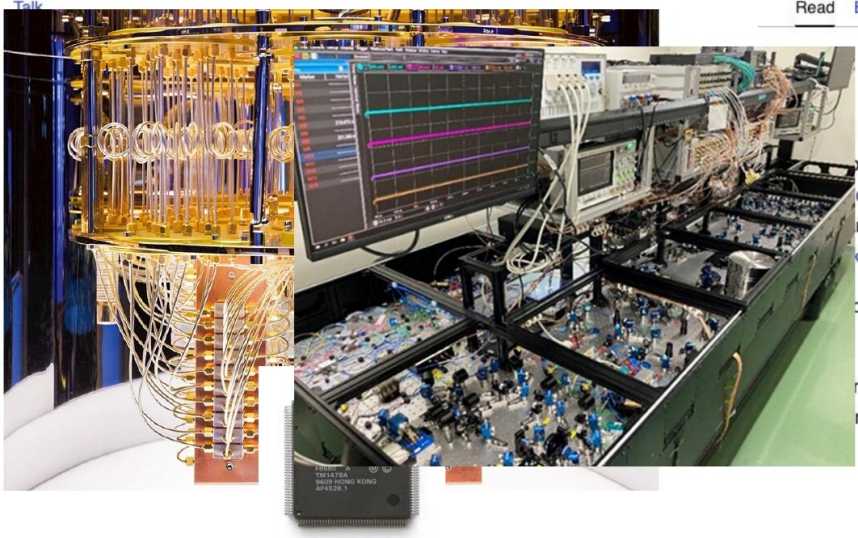
ubit quantum processor.[9] However,
ssors still remaining the norm.[10]

Noisy intermediate-scale quantum computing

7 languages

Article Talk

Read Edit View history Tools



up to 1,000 qubits which
processors, which are
quantum error correction.
nd gate fidelity. The
kill in 2018.[6][4]

quantum computing

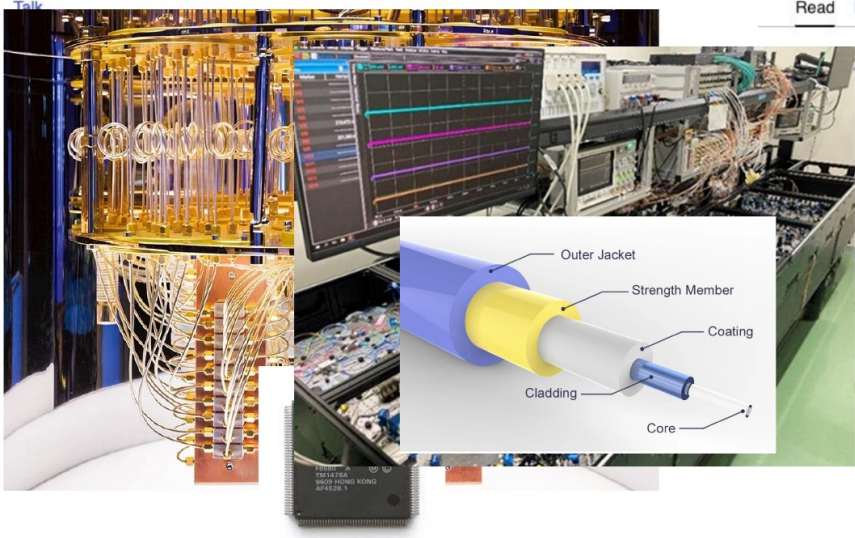
m processor.[9] However,
maining the norm.[10]

Noisy intermediate-scale quantum computing

7 languages

Article Talk

Read Edit View history Tools



up to 1,000 qubits which
processors, which are
quantum error correction.
nd gate fidelity. The
kill in 2018.[6][4]

quantum computing

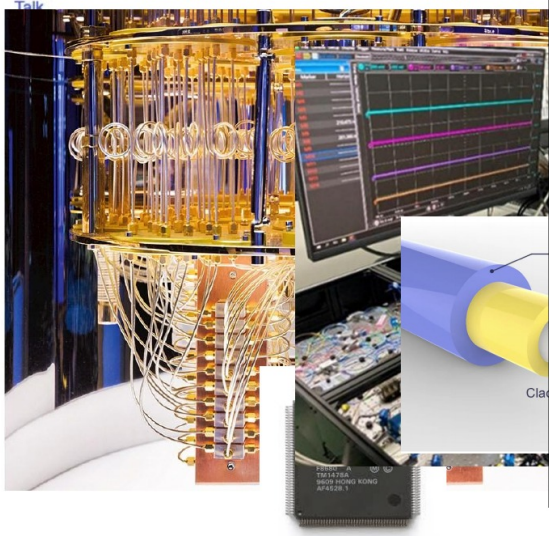
m processor.[9] However,
maining the norm.[10]

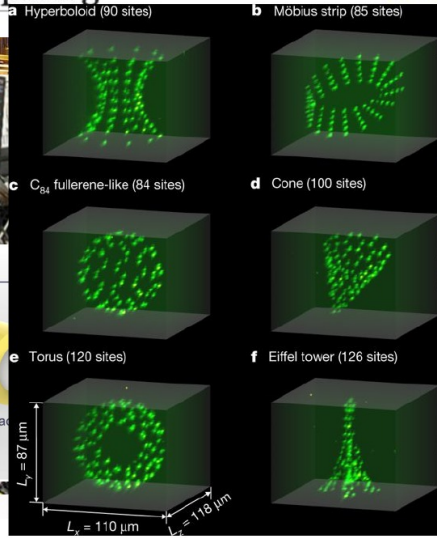
Noisy intermediate-scale quantum computing

7 languages

Article Talk

Tools





bits which
ich are
correction.
. The

uting

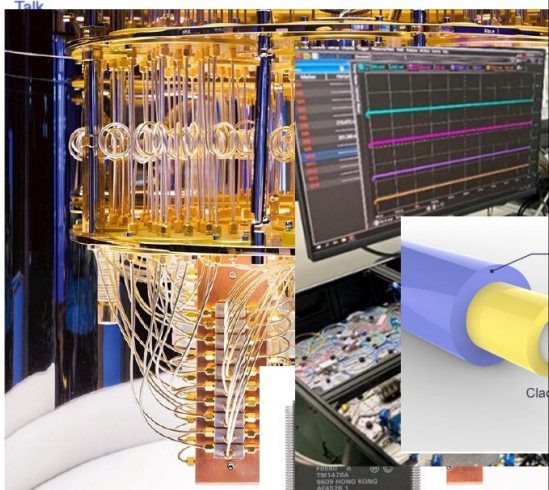
However,
rm.[10]

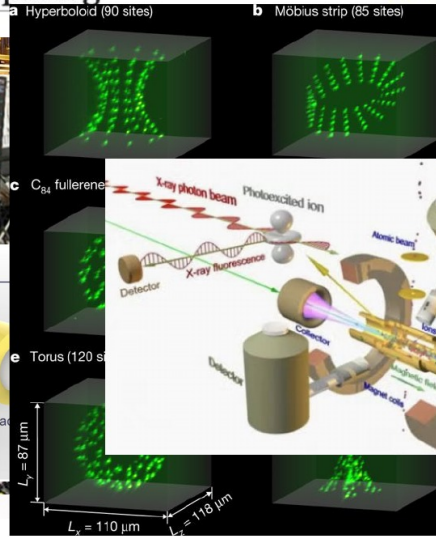
Noisy intermediate-scale quantum computing

7 languages

Article Talk

Tools





Quantum Physics

[Submitted on 10 Feb 2025]

A Review and Collection of Metrics and Benchmarks for Quantum Computers: definitions, methodologies and software

Deep Lall, Abhishek Agarwal, Weixi Zhang, Lachlan Lindoy, Tobias Lindström, Stephanie Webster, Simon Hall, Nicholas Chancellor, Petros Wallden, Raul Garcia-Patron, Elham Kashefi, Viv Kendon, Jonathan Pritchard, Alessandro Rossi, Animesh Datta, Theodoros Kapourniotis, Konstantinos Georgopoulos, Ivan Rungger

are not advanced enough yet for fault-tolerance or large enough to achieve quantum advantage.[1][2] These processors, which are sensitive to their environment (noisy) and prone to quantum decoherence, are not yet capable of continuous quantum error correction. This intermediate-scale is defined by the quantum volume, which is based on a moderate number of qubits and gate fidelity. The **NISQ era** is the current state of quantum computer technology,[3][4][5] and the term was coined by John Preskill in 2018.[6][4]

According to Microsoft Azure Quantum's scheme, NISQ computation is considered level 1, the lowest of the quantum computing implementation levels.[7][8]

In October 2023, the 1,000 qubit mark was passed for the first time by Atom Computing's 1,180 qubit quantum processor.[9] However, as of 2024, only two quantum processors have over 1,000 qubits, with sub-1,000 quantum processors still remaining the norm.[10]

How to Build a Quantum Supercomputer: Scaling from Hundreds to Millions of Qubits

Masoud Mohseni, Artur Scherer, K. Grace Johnson, Oded Wertheim, Matthew Otten, Namit Anand, Navid Anjum Aadit, Yuri Alexeev, Gilad Ben-Shach, Kirk M. Bresniker, Kerem Y. Camsari, Barbara Chapman, Soumitra Chatterjee, Shuvro Chowdhury, Gebremedhin A. Dagnew, Tom Dvir, Aniello Esposito, Farah Fahim, Michael Ferguson, Marco Fiorentino, Archit Gajjar, Katerina Gratsea, Gaurav Gyawali, Christian Heiter, Ali H. Z. Kavaki, Abdullah Khalid, Xiangzhou Kong, Bohdan Kulchytskyy, Elica Kyoseva, Ruoyu Li, P. Aaron Lott, Igor L. Markov, Robert F. McDermott, Lucas Morais, Giacomo Pedretti, Pooja Rao, Eleanor Rieffel, Allyson Silva, John Sorebo, Panagiotis Spentzouris, Ziv Steiner, Boyan Torosov, Davide Venturelli, Robert J. Visser, Zak Webb, Xin Zhan, Yonatan Cohen, Pooya Ronagh, Alan Ho, Raymond G. Beausoleil, John M. Martinis

...van

...error correction.

...number of qubits and gate fidelity. The

...and the term was coined by John Preskill in 2018.[6][4]

...Microsoft Azure Quantum's scheme, NISQ computation is considered level 1, the lowest of the quantum computing implementation levels.[7][8]

In October 2023, the 1,000 qubit mark was passed for the first time by Atom Computing's 1,180 qubit quantum processor.[9] However, as of 2024, only two quantum processors have over 1,000 qubits, with sub-1,000 quantum processors still remaining the norm.[10]

Quantum Physics

[Submitted on 10 Feb 2025]

Quantum Physics

[Submitted on 12 Feb 2026 (v1), last revised 5 May 2026 (this version, v2)]

The Pinnacle Architecture: Reducing the cost of breaking RSA-2048 to 100 000 physical qubits using quantum LDPC codes

Paul Webster, Lucas Berent, Omprakash Chandra, Evan T. Hockings, Nouédyn Baspin, Felix Thomsen, Samuel C. Smith, Lawrence Z. Cohen
Allyson
Ronagh, Alan
Microsoft

[Submitted on 13 Mar 2026 (this version, v3)]

Quantum Supercomputer: Scaling from Hundreds to Millions of Qubits

Johnson, Oded Wertheim, Matthew Otten, Namit Anand, Navid Anjum Aadit, Yuri Alexeev, Gilad Ben-Shach, Kirk M.
Soumitra Chatterjee, Shuvro Chowdhury, Gebremedhin A. Dagnew, Tom Dvir, Aniello Esposito, Farah Fahim,
rina Gratsea, Gaurav Gyawali, Christian Heiter, Ali H. Z. Kavaki, Abdullah Khalid, Xiangzhou Kong,
L. Markov, Robert F. McDermott, Lucas Morais, Giacomo Pedretti, Pooja Rao, Eleanor Rieffel,
Torosov, Davide Venturelli, Robert J. Visser, Zak Webb, Xin Zhan, Yonatan Cohen, Pooya

...van

...quantum error correction.

...ate number of qubits and gate fidelity. The

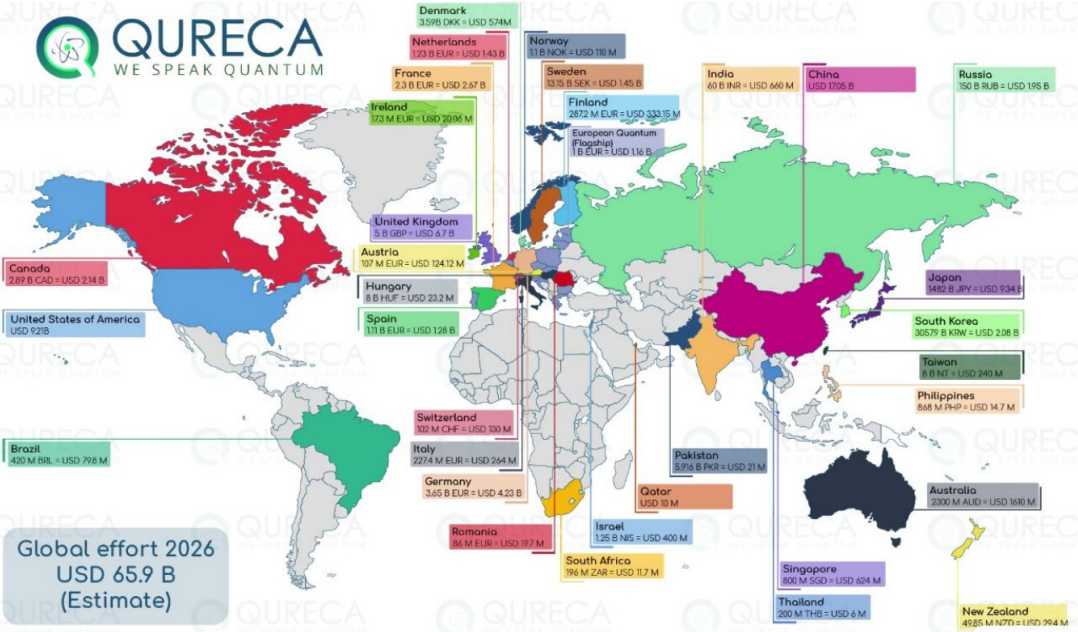
...e term was coined by John Preskill in 2018.[6][4]

...ost of the quantum computing

implementation levels.[7][8]

In October 2023, the 1,000 qubit mark was passed for the ...
as of 2024, only two quantum processors have over 1,000 qubits, with su...

QURECA
WE SPEAK QUANTUM



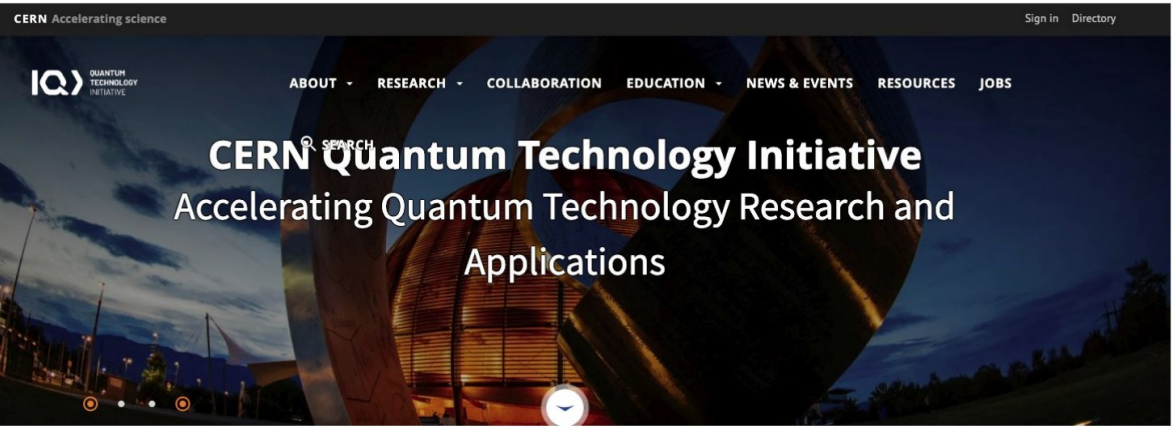
Global effort 2026
USD 65.9 B
(Estimate)


Versailles
Parigi
Strasburgo
Stoccarda
Slovacchia
Košice
Rennes
Le Mans
Linz
Vienna
Monaco di Baviera
Friburgo in Brisgovia
Salisburgo
Austria
Budapest
Debrecen
Angers
Tours
Basilea
Zurigo
Innsbruck
Nantes
Digione
Graz
Ungheria
Francia
Svizzera
Losanna
Poitiers
La Rochelle
Ginevra
Seghedino
Arad
Slovenia
Clermont-Ferrand
Lione
Zagabria
Timişoara
Bergamo
Trieste
Milano
Verona
Venezia
Novi Sad
Нови Сад
Grenoble
Torino
Croazia
Bordeaux
Pola
Belgrado
Београд
Parma
Bosnia ed Erzegovina
Genova
Bologna
Zara
Serbia
Avignone
San Marino
Sarajevo
Tolosa
Montpellier
Monaco
Firenze
Spalato
Sebastián
Carcassonne
Marsiglia
Mostar
Macarsca
Niš
Ниш
Italia
Pamplona
Montenegro
Perpignano
Andorra
Mare Adriatico
Kosovo
Pescara
Podgorica
Подгорица
Girona
Corse
Roma
Saragozza
Lloret de Mar
Macedonia del Nord
Terrassa
Barcellona
Tirana
Tarragona
Bari
Albania
Ocrida
Охрид
Olbia
Napoli
Sassari
Salerno
Matera


Versailles
Parigi
Stoccarda
Strasburgo
Slovacchia
Košice
Rennes
Le Mans
Friburgo in Brisgovia
Monaco di Baviera
Linz
Vienna
Salisburgo
Austria
Budapest
Debrecen
Angers
Tours
Nantes
Innsbruck
Graz
Ungheria
Poitiers
La Rochelle
Seghedino
Arad
Slovenia
Zagabria
Timişoara
Verona
Venezia
Trieste
Novi Sad
Нови Сад
Croazia
Pola
Bordeaux
Belgrado
Београд
Bosnia ed Erzegovina
Bologna
Zara
Serbia
Avignone
Monaco
San Marino
Firenze
Sarajevo
Tolosa
Montpellier
Spalato
Mostar
Sebastián
Carcassonne
Marsiglia
Macarsca
Niš
Ниш
Pamplona
Perpignano
Mare Adriatico
Montenegro
Andorra
Podgorica
Подгорица
Kosovo
Girona
Corse
Saragozza
Lloret de Mar
Macedonia del Nord
Terrassa
Barcellona
Tirana
Tarragona
Albania
Ocrida
Охрид
Olbia
Bari
Sassari
Napoli
Salerno
Matera
Genève
CERN
ATLAS
CMS
ALICE
LHC – 27 km
SAPIENZA
UNIVERSITÀ DI ROMA


Monaco

Versailles
Parigi
Strasburgo
Stoccarda
Monaco di Baviera
Friburgo in Brisgovia
Linz
Vienna
Slovacchia
Košice
Rennes
Le Mans
Salisburgo
Austria
Budapest
Debrecen
Angers
Tours
Innsbruck
Graz
Ungheria
Nantes
Poitiers
La Rochelle
Seghedino
Arad
Slovenia
Zagabria
Timişoara
Verona
Venezia
Trieste
Novi Sad
Нови Сад
Croazia
Pola
Bordeaux
Belgrado
Београд
Bosnia ed Erzegovina
Bologna
Zara
Serbia
San Marino
Firenze
Sarajevo
Tolosa
Spalato
Mostar
Niš
Ниш
Sebastián
Carcassonne
Macarsca
Italia
Montenegro
Pamplona
Perpignano
Andorra
Mare Adriatico
Kosovo
Pescara
Podgorica
Подгорица
Girona
Corse
Roma
Saragozza
Lloret de Mar
Terrassa
Barcellona
Macedonia del Nord
Tirana
Tarragona
Bari
Albania
Ocrida
Охрид
Olbia
Napoli
Sassari
Salerno
Matera


CERN Accelerating science
Sign in
Directory
QUANTUM TECHNOLOGY INITIATIVE
ABOUT
RESEARCH
COLLABORATION
EDUCATION
NEWS & EVENTS
RESOURCES
JOBS
SEARCH
CERN Quantum Technology Initiative
Accelerating Quantum Technology Research and Applications

Leonardo Lavagna

- Sapienza University
- Email
- ResearchGate
- LinkedIn
- Github
- HackerRank
- Kaggle
- Google Scholar

https://leonardolavagna.github.io/

“The Talk”

- **Overview**
- **Interactive Discussion on Quantum Computing and Algorithms**

Bits, Qubits, (Quantum) Gates, (Quantum) Turing Machines, and Public key cryptography

How many of you have done programming?

Bits, Qubits, (Quantum) Gates, (Quantum) Turing Machines, and Public key cryptography

How many of you have done programming?

We will not need an actual “quantum” computer.

Bits, Qubits, (Quantum) Gates, (Quantum) Turing Machines, and Public key cryptography

How many of you have done programming?

We will not need an actual "quantum" computer.
Instead, (quantum) computer science:

- is not exactly a science only (art, simulated experiments);

Bits, Qubits, (Quantum) Gates, (Quantum) Turing Machines, and Public key cryptography

How many of you have done programming?

We will not need an actual "quantum" computer.
Instead, (quantum) computer science:

- is not exactly a science only (art, simulated experiments);
- is not about computers and history, just as astronomy is not only about telescopes and particle physics is not only about colliders.

Bits, Qubits, (Quantum) Gates, (Quantum) Turing Machines, and Public key cryptography

How many of you have done programming?

We will not need an actual "quantum" computer.
Instead, (quantum) computer science:

- is not exactly a science only (art, simulated experiments);
- is not about computers and history, just as astronomy is not only about telescopes and particle physics is not only about colliders.

Bits, Qubits, (Quantum) Gates, (Quantum) Turing Machines, and Public key cryptography

How many of you have done programming?

We will not need an actual "quantum" computer.
Instead, (quantum) computer science:

- is not exactly a science only (art, simulated experiments);
- is not about computers and history, just as astronomy is not only about telescopes and particle physics is not only about colliders.

All we need is a way to process information.

Bits, Qubits, (Quantum) Gates, (Quantum) Turing Machines, and Public key cryptography

How many of you have done programming?

We will not need an actual "quantum" computer.
Instead, (quantum) computer science:

- is not exactly a science only (art, simulated experiments);
- is not about computers and history, just as astronomy is not only about telescopes and particle physics is not only about colliders.

All we need is a way to process information.

What is information? Think of a Morse message.

Bits and bit strings

The basic unit of information is the bit: $\{0, 1\}$.

Bits and bit strings

The basic unit of information is the bit: $\{0, 1\}$.

We can represent "knowledge" as strings of bits:

$$x_1, x_2, \ldots, x_n \in \{0, 1\}.$$

Bits and bit strings

The basic unit of information is the bit: $\{0, 1\}$.

We can represent "knowledge" as strings of bits:

$$x_1, x_2, \ldots, x_n \in \{0, 1\}.$$

Example:

$$01100 \ldots 101.$$

Bits and bit strings

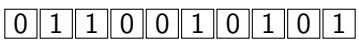
The basic unit of information is the bit: $\{0, 1\}$.

We can represent "knowledge" as strings of bits:

$$x_1, x_2, \ldots, x_n \in \{0, 1\}.$$

Example:

$$01100\ldots101.$$



Exercise: how many strings?

If we have n bits, how many strings can we write?

Exercise: how many strings?

If we have n bits, how many strings can we write?

number of bits	strings	how many
1	$0, 1$	2
2	$00, 01, 10, 11$	2^2
3	$000, 001, 010, 011, 100, 101, 110, 111$	2^3
n	$x_1 \ldots x_n$	2^n

Exercise: how many strings?

If we have n bits, how many strings can we write?

number of bits	strings	how many
1	$0, 1$	2
2	$00, 01, 10, 11$	2^2
3	$000, 001, 010, 011, 100, 101, 110, 111$	2^3
n	$x_1 \dots x_n$	2^n

Examples of scale:

- $2^{10} \simeq 10^3$;

Exercise: how many strings?

If we have n bits, how many strings can we write?

number of bits	strings	how many
1	$0, 1$	2
2	$00, 01, 10, 11$	2^2
3	$000, 001, 010, 011, 100, 101, 110, 111$	2^3
n	$x_1 \dots x_n$	2^n

Examples of scale:

- $2^{10} \simeq 10^3$;
- $2^{32} \simeq 4 \cdot 10^9$;

Exercise: how many strings?

If we have n bits, how many strings can we write?

number of bits	strings	how many
1	$0, 1$	2
2	$00, 01, 10, 11$	2^2
3	$000, 001, 010, 011, 100, 101, 110, 111$	2^3
n	$x_1 \dots x_n$	2^n

Examples of scale:

- $2^{10} \simeq 10^3$;
- $2^{32} \simeq 4 \cdot 10^9$;
- $2^{300} \sim 10^{90}$.

Exercise: how many strings?

If we have n bits, how many strings can we write?

number of bits	strings	how many
1	$0, 1$	2
2	$00, 01, 10, 11$	2^2
3	$000, 001, 010, 011, 100, 101, 110, 111$	2^3
n	$x_1 \dots x_n$	2^n

Examples of scale:

- $2^{10} \simeq 10^3$;
- $2^{32} \simeq 4 \cdot 10^9$;
- $2^{300} \sim 10^{90}$.

Exercise: how many strings?

If we have n bits, how many strings can we write?

number of bits	strings	how many
1	$0, 1$	2
2	$00, 01, 10, 11$	2^2
3	$000, 001, 010, 011, 100, 101, 110, 111$	2^3
n	$x_1 \dots x_n$	2^n

Examples of scale:

- $2^{10} \simeq 10^3$;
- $2^{32} \simeq 4 \cdot 10^9$;
- $2^{300} \sim 10^{90}$.

One byte $= 8$ bits, $1\,\mathrm{kB} = 2^{10}$ bytes, $1\,\mathrm{MB} = 2^{20}$ bytes, $1\,\mathrm{GB} = 2^{30}$ bytes.

Encodings

Of course, with different encodings we could work with trits $\{0, 1, 2\}$, integers, and so on.

Encodings

Of course, with different encodings we could work with trits $\{0, 1, 2\}$, integers, and so on.

For classical computers it is convenient to use bits $\{0, 1\}$, also called "on/off".

Encodings

Of course, with different encodings we could work with trits $\{0, 1, 2\}$, integers, and so on.

For classical computers it is convenient to use bits $\{0, 1\}$, also called "on/off".

Exercise: encode the word "LEO".

$$A = 00000, \quad B = 00001, \quad C = 00010, \quad D = 00011,$$
$$E = 00100, \quad \ldots, \quad L = 01011, \quad \ldots, \quad O = 01110.$$

Encodings

Of course, with different encodings we could work with trits $\{0, 1, 2\}$, integers, and so on.

For classical computers it is convenient to use bits $\{0, 1\}$, also called "on/off".

Exercise: encode the word "LEO".

$$A = 00000, \quad B = 00001, \quad C = 00010, \quad D = 00011,$$
$$E = 00100, \quad \ldots, \quad L = 01011, \quad \ldots, \quad O = 01110.$$

Therefore:

$$\text{"LEO"} = 01011\ 00100\ 01110.$$

Encodings

Of course, with different encodings we could work with trits $\{0, 1, 2\}$, integers, and so on.

For classical computers it is convenient to use bits $\{0, 1\}$, also called "on/off".

Exercise: encode the word "LEO".

$$A = 00000, \quad B = 00001, \quad C = 00010, \quad D = 00011,$$
$$E = 00100, \quad \ldots, \quad L = 01011, \quad \ldots, \quad O = 01110.$$

Therefore:

$$\text{"LEO"} = 01011\ 00100\ 01110.$$

ASCII uses 8 bits per character.

Exercise: encode a 2×2 RGB image

A 2×2 image has four pixels. Each RGB channel can take values in $\{0, \ldots, 255\}$.

Exercise: encode a 2×2 RGB image

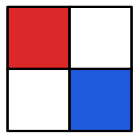
A 2×2 image has four pixels. Each RGB channel can take values in $\{0, \ldots, 255\}$.

$$
\begin{aligned}
\text{red} &= 11111111\ 00000000\ 00000000, \\
\text{blue} &= 00000000\ 00000000\ 11111111, \\
\text{white} &= 11111111\ 11111111\ 11111111.
\end{aligned}
$$

Exercise: encode a 2×2 RGB image

A 2×2 image has four pixels. Each RGB channel can take values in $\{0, \ldots, 255\}$.

$$
\begin{aligned}
\text{red} &= 11111111\ 00000000\ 00000000, \\
\text{blue} &= 00000000\ 00000000\ 11111111, \\
\text{white} &= 11111111\ 11111111\ 11111111.
\end{aligned}
$$



box / image

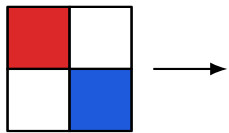
(red, white, white, blue)
=(11111111 00000000 00000000,
11111111 11111111 11111111,
11111111 11111111 11111111,
00000000 00000000 11111111)

vector

Exercise: encode a 2×2 RGB image

A 2×2 image has four pixels. Each RGB channel can take values in $\{0, \ldots, 255\}$.

$$
\begin{aligned}
\text{red} &= 11111111\ 00000000\ 00000000, \\
\text{blue} &= 00000000\ 00000000\ 11111111, \\
\text{white} &= 11111111\ 11111111\ 11111111.
\end{aligned}
$$



box / image

(red, white, white, blue)
=(11111111 00000000 00000000,
11111111 11111111 11111111,
11111111 11111111 11111111,
00000000 00000000 11111111)

vector

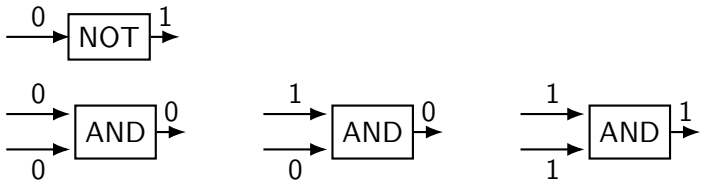
With a computer we want to encode information but also to process it: classical computing.

Manipulating information

We manipulate information using combinations of gates transforming bit strings.

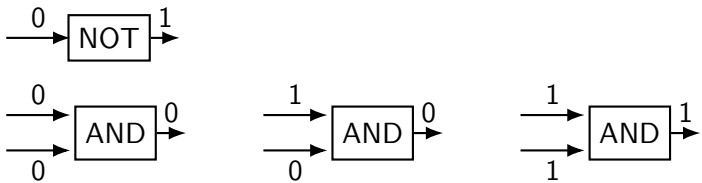
Manipulating information

We manipulate information using combinations of gates transforming bit strings.



Manipulating information

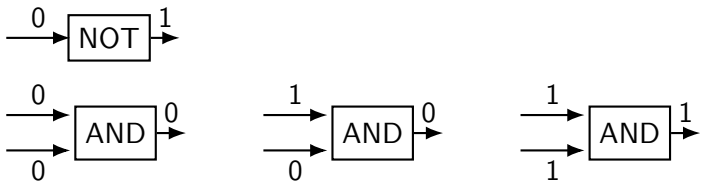
We manipulate information using combinations of gates transforming bit strings.



Examples of gates: NOT, AND, OR, XOR, NAND.

Manipulating information

We manipulate information using combinations of gates transforming bit strings.



Examples of gates: NOT, AND, OR, XOR, NAND.

These gates have tabular representations.

Truth tables

Truth tables

NOT:

x	$\mathrm{NOT}(x)$
0	1
1	0

Truth tables

NOT:

x	$\mathrm{NOT}(x)$
0	1
1	0

AND:

x	y	$\mathrm{AND}(x, y)$
0	0	0
0	1	0
1	0	0
1	1	1

Truth tables

NOT:

x	$\mathrm{NOT}(x)$
0	1
1	0

AND:

x	y	$\mathrm{AND}(x, y)$
0	0	0
0	1	0
1	0	0
1	1	1

Truth tables

NOT:

x	$\mathrm{NOT}(x)$
0	1
1	0

AND:

x	y	$\mathrm{AND}(x,y)$
0	0	0
0	1	0
1	0	0
1	1	1

A classical process can be represented as a Boolean function, i.e. a composition of gates.

Example:

$$f(x,y) = \mathrm{NOT}(\mathrm{AND}(x,y)) : \{0,1\}^2 \to \{0,1\}.$$

Truth tables

NOT:

x	$\mathrm{NOT}(x)$
0	1
1	0

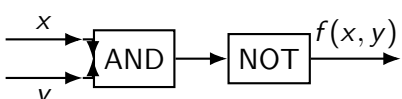
AND:

x	y	$\mathrm{AND}(x,y)$
0	0	0
0	1	0
1	0	0
1	1	1

A classical process can be represented as a Boolean function, i.e. a composition of gates.

Example:

$$f(x,y) = \mathrm{NOT}(\mathrm{AND}(x,y)) : \{0,1\}^2 \to \{0,1\}.$$



Boolean functions

In general:

- Boolean functions are not reversible. Why?

Boolean functions

In general:

- Boolean functions are not reversible. Why?
- Boolean generators are deterministic. Why?

Boolean functions

In general:

- Boolean functions are not reversible. Why?
- Boolean generators are deterministic. Why?

Boolean functions

In general:

- Boolean functions are not reversible. Why?
- Boolean generators are deterministic. Why?

This gives constraints on classical algorithms.

Boolean functions

In general:

- Boolean functions are not reversible. Why?
- Boolean generators are deterministic. Why?

This gives constraints on classical algorithms.

They are not solved by quantum algorithms.

Boolean functions

In general:

- Boolean functions are not reversible. Why?
- Boolean generators are deterministic. Why?

This gives constraints on classical algorithms.

They are not solved by quantum algorithms.

Exercise: build a Boolean function that realizes a simple comparison test.

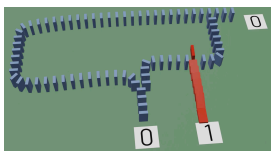
From transistors to chips

One of the greatest achievements of the last century is the transistor.

From transistors to chips

One of the greatest achievements of the last century is the transistor.

A transistor makes it easy to build gates.



From transistors to chips

One of the greatest achievements of the last century is the transistor.

A transistor makes it easy to build gates.

Today:

- a classical chip has billions of transistors;

From transistors to chips

One of the greatest achievements of the last century is the transistor.

A transistor makes it easy to build gates.

Today:

- a classical chip has billions of transistors;
- there is no standardized “quantum transistor”.

Universality of AND and NOT

Theorem: AND and NOT are universal.

Universality of AND and NOT

Theorem: AND and NOT are universal.

This gives a way to realize any Boolean function.

Universality of AND and NOT

Theorem: AND and NOT are universal.

This gives a way to realize any Boolean function.

Universality of AND and NOT

Theorem: AND and NOT are universal.

This gives a way to realize any Boolean function.

Why is this not enough?

Universality of AND and NOT

Theorem: AND and NOT are universal.

This gives a way to realize any Boolean function.

Why is this not enough?

Because any circuit still has to work with a fixed number of bits.

Exercise: a program

A program to compute $m!$:

Exercise: a program

A program to compute $m!$:

input	m
f	$\leftarrow 1$
for $i = 1$ to m	$f \leftarrow f \cdot i$
return	f

Exercise: a program

A program to compute $m!$:

input	m
f	$\leftarrow 1$
for $i = 1$ to m	$f \leftarrow f \cdot i$
return	f

It works for every m.

Exercise: a program

A program to compute $m!$:

input	m
f	$\leftarrow 1$
for $i = 1$ to m	$f \leftarrow f \cdot i$
return	f

It works for every m.

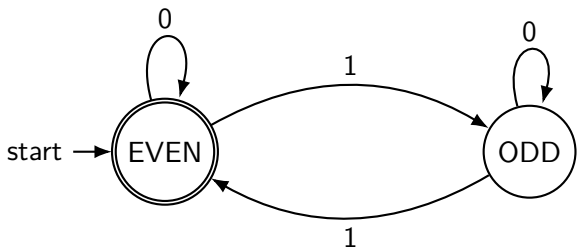
It works for every m if it fits the memory.

Automata

Automata read any bit string, maintaining an internal "state" and updating the state according to some rule.

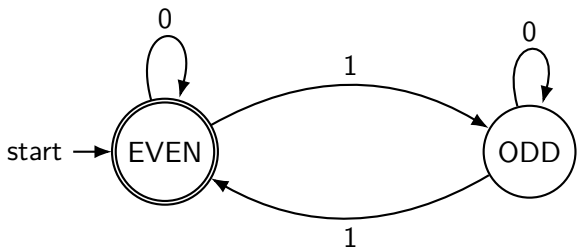
Automata

Automata read any bit string, maintaining an internal "state" and updating the state according to some rule.



Automata

Automata read any bit string, maintaining an internal "state" and updating the state according to some rule.



Exercise: design an automaton for strings with an odd number of ones.

Regular expressions

A regular expression is a string of bits where one can use parentheses, Boolean operations, and repetition.

Regular expressions

A regular expression is a string of bits where one can use parentheses, Boolean operations, and repetition.

For instance, for strings built out of zeros and ones, we can use:

- parentheses;

Regular expressions

A regular expression is a string of bits where one can use parentheses, Boolean operations, and repetition.

For instance, for strings built out of zeros and ones, we can use:

- parentheses;
- OR;

Regular expressions

A regular expression is a string of bits where one can use parentheses, Boolean operations, and repetition.

For instance, for strings built out of zeros and ones, we can use:

- parentheses;
- OR;
- the star *, meaning “repeat”.

Regular expressions

A regular expression is a string of bits where one can use parentheses, Boolean operations, and repetition.

For instance, for strings built out of zeros and ones, we can use:

- parentheses;
- OR;
- the star *, meaning “repeat”.

Regular expressions

A regular expression is a string of bits where one can use parentheses, Boolean operations, and repetition.

For instance, for strings built out of zeros and ones, we can use:

- parentheses;
- OR;
- the star *, meaning “repeat”.

Regular expressions are the bridge between automata and Boolean circuits.

Regular expressions

A regular expression is a string of bits where one can use parentheses, Boolean operations, and repetition.

For instance, for strings built out of zeros and ones, we can use:

- parentheses;
- OR;
- the star *, meaning "repeat".

Regular expressions are the bridge between automata and Boolean circuits.

Theorem: a set of strings has a regular expression if and only if it has an automaton.

Regular expressions

A regular expression is a string of bits where one can use parentheses, Boolean operations, and repetition.

For instance, for strings built out of zeros and ones, we can use:

- parentheses;
- OR;
- the star *, meaning "repeat".

Regular expressions are the bridge between automata and Boolean circuits.

Theorem: a set of strings has a regular expression if and only if it has an automaton.

Exercise: define a regular expression for the set of strings with an even number of ones.

Why automata are not enough

Now, regular expressions and automata are “weak”.

Why automata are not enough

Now, regular expressions and automata are “weak”.

For instance, they cannot say if a string is a palindrome.

Why automata are not enough

Now, regular expressions and automata are “weak”.

For instance, they cannot say if a string is a palindrome.

Is a string a palindrome?

Why automata are not enough

Now, regular expressions and automata are “weak”.

For instance, they cannot say if a string is a palindrome.

Is a string a palindrome?

Why is this not enough?

Because automata have finite memory.

Why automata are not enough

Now, regular expressions and automata are “weak”.

For instance, they cannot say if a string is a palindrome.

Is a string a palindrome?

Why is this not enough?

Because automata have finite memory.

For the previous question, one may need to “remember” the first part of a string.

Why automata are not enough

Now, regular expressions and automata are “weak”.

For instance, they cannot say if a string is a palindrome.

Is a string a palindrome?

Why is this not enough?

Because automata have finite memory.

For the previous question, one may need to “remember” the first part of a string.

Turing machines are the computing model that puts together:

- Boolean circuits;

Why automata are not enough

Now, regular expressions and automata are “weak”.

For instance, they cannot say if a string is a palindrome.

Is a string a palindrome?

Why is this not enough?

Because automata have finite memory.

For the previous question, one may need to “remember” the first part of a string.

Turing machines are the computing model that puts together:

- Boolean circuits;
- finite automata / regular expressions;

Why automata are not enough

Now, regular expressions and automata are “weak”.

For instance, they cannot say if a string is a palindrome.

Is a string a palindrome?

Why is this not enough?

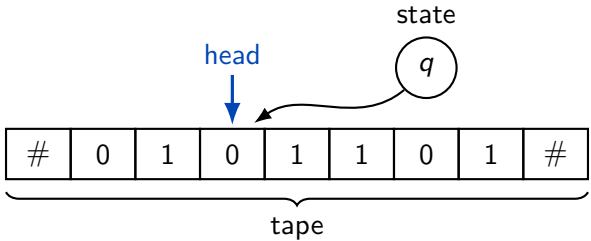
Because automata have finite memory.

For the previous question, one may need to “remember” the first part of a string.

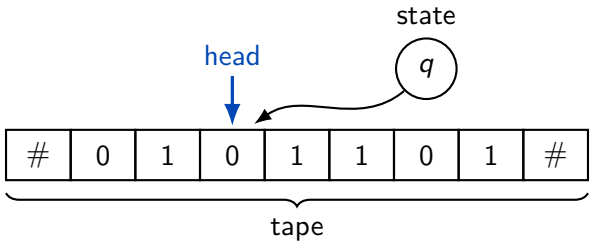
Turing machines are the computing model that puts together:

- Boolean circuits;
- finite automata / regular expressions;
- infinite memory.

Turing machines

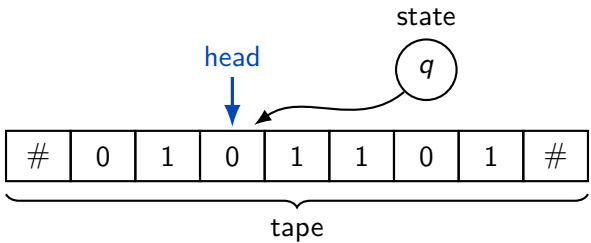


Turing machines



A Turing machine uses a tape, a head and a finite internal state.

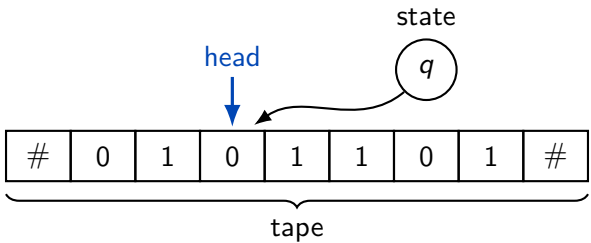
Turing machines



A Turing machine uses a tape, a head and a finite internal state.

There are Turing machines for AND, NOT, $1 + 1 = 2$, for any regular expression, and so on.

Turing machines



A Turing machine uses a tape, a head and a finite internal state.

There are Turing machines for AND, NOT, $1 + 1 = 2$, for any regular expression, and so on.

Exercise: Turing machine for AND(x, y).

Turing machines and halting

Exercise: check out the halting problem and related topics.

Turing machines and halting

Exercise: check out the halting problem and related topics.

This is the “low level” description of any software implemented in a classical hardware whose unit of information is the bit.

Turing machines and halting

Exercise: check out the halting problem and related topics.

This is the "low level" description of any software implemented in a classical hardware whose unit of information is the bit.

Let us discuss the unit of quantum information and the relevant gates / automata and Turing machines.

Cryptographic implications

We have seen how quantum computers are essentially different from classical processors.

Cryptographic implications

We have seen how quantum computers are essentially different from classical processors.

Let us conclude with cryptographic implications.

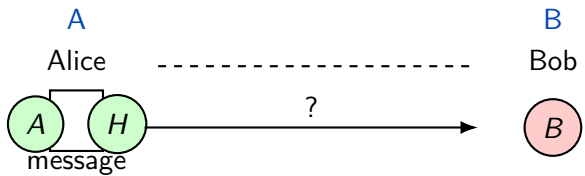
Cryptographic implications

We have seen how quantum computers are essentially different from classical processors.

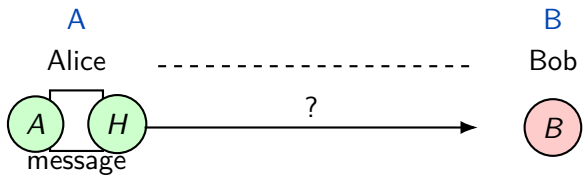
Let us conclude with cryptographic implications.

Of such differences, consider an asymmetric example common in peer-to-peer internet communications.

Alice and Bob

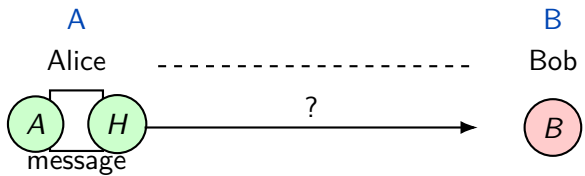


Alice and Bob



A sends a message to B through a public channel.

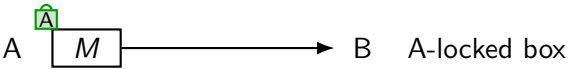
Alice and Bob



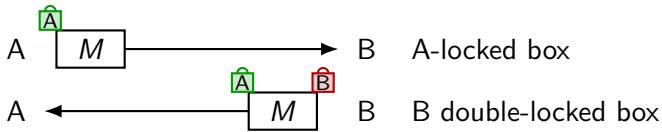
A sends a message to B through a public channel.

The double-lock trick gives the idea.

The double-lock trick

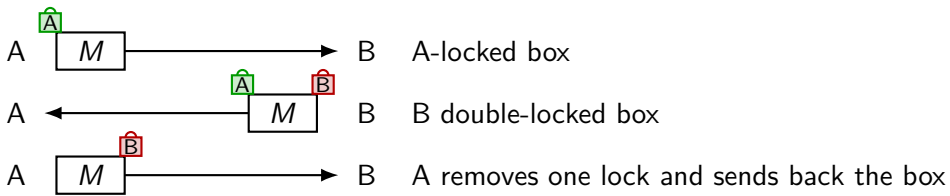


The double-lock trick



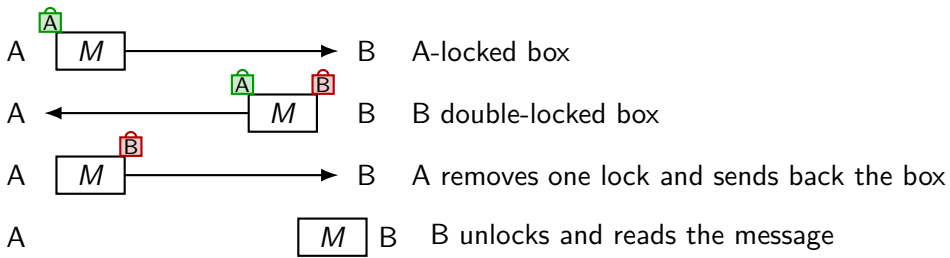
The mathematical protocol that implements this trick, for example RSA, is based on the "lock" number and the "key" number.

The double-lock trick



The mathematical protocol that implements this trick, for example RSA, is based on the "lock" number and the "key" number.

The double-lock trick



The mathematical protocol that implements this trick, for example RSA, is based on the "lock" number and the "key" number.

Clock arithmetic

"Clock" arithmetic is arithmetic modulo n.

Clock arithmetic

"Clock" arithmetic is arithmetic modulo n.

Examples:

- on a clock with 12 hours: $1 = 13$, $2 = 14$, ...;

Clock arithmetic

"Clock" arithmetic is arithmetic modulo n.

Examples:

- on a clock with 12 hours: $1 = 13$, $2 = 14$, ...;
- $1 + 2 = 3$;

Clock arithmetic

"Clock" arithmetic is arithmetic modulo n.

Examples:

- on a clock with 12 hours: $1 = 13$, $2 = 14$, ...;
- $1 + 2 = 3$;
- $3 \cdot 5 = 15 = 3$ modulo 12;

Clock arithmetic

"Clock" arithmetic is arithmetic modulo n.

Examples:

- on a clock with 12 hours: $1 = 13$, $2 = 14$, ...;
- $1 + 2 = 3$;
- $3 \cdot 5 = 15 = 3$ modulo 12;
- $2 + 7 = 9$ modulo 12;

Clock arithmetic

"Clock" arithmetic is arithmetic modulo n.

Examples:

- on a clock with 12 hours: $1 = 13$, $2 = 14$, ...;
- $1 + 2 = 3$;
- $3 \cdot 5 = 15 = 3$ modulo 12;
- $2 + 7 = 9$ modulo 12;
- $39 = 3$ modulo 12.

Clock arithmetic

"Clock" arithmetic is arithmetic modulo n.

Examples:

- on a clock with 12 hours: $1 = 13$, $2 = 14$, ...;
- $1 + 2 = 3$;
- $3 \cdot 5 = 15 = 3$ modulo 12;
- $2 + 7 = 9$ modulo 12;
- $39 = 3$ modulo 12.

Clock arithmetic

"Clock" arithmetic is arithmetic modulo n.

Examples:

- on a clock with 12 hours: $1 = 13$, $2 = 14$, ...;
- $1 + 2 = 3$;
- $3 \cdot 5 = 15 = 3$ modulo 12;
- $2 + 7 = 9$ modulo 12;
- $39 = 3$ modulo 12.

In this arithmetic we can define a classically hard problem to solve with a Turing machine.

RSA: the hard direction

Let

$$n = p \cdot q$$

be the product of large primes.

RSA: the hard direction

Let

$$n = p \cdot q$$

be the product of large primes.

Euler's function:

$$\varphi(n) = (p-1)(q-1).$$

RSA: the hard direction

Let

$$n = p \cdot q$$

be the product of large primes.

Euler's function:

$$\varphi(n) = (p-1)(q-1).$$

It is easy to go

$$(p, q) \longrightarrow p \cdot q = n.$$

RSA: the hard direction

Let

$$n = p \cdot q$$

be the product of large primes.

Euler's function:

$$\varphi(n) = (p-1)(q-1).$$

It is easy to go

$$(p, q) \longrightarrow p \cdot q = n.$$

It is hard to go

$$n \longrightarrow p, q.$$

RSA: public and private numbers

RSA: public and private numbers

A

RSA: public and private numbers

A

- chooses p, q secret;

RSA: public and private numbers

A

- chooses p, q secret;
- $n = pq$;

RSA: public and private numbers

A

- chooses p, q secret;
- $n = pq$;
- $\varphi(n) = (p-1)(q-1)$;

RSA: public and private numbers

A

- chooses p, q secret;
- $n = pq$;
- $\varphi(n) = (p-1)(q-1)$;
- chooses e with $\gcd(e, \varphi(n)) = 1$;

RSA: public and private numbers

A

- chooses p, q secret;
- $n = pq$;
- $\varphi(n) = (p-1)(q-1)$;
- chooses e with $\gcd(e, \varphi(n)) = 1$;
- finds d with $ed = 1 \bmod \varphi(n)$;

RSA: public and private numbers

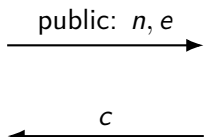
A

- chooses p, q secret;
- $n = pq$;
- $\varphi(n) = (p-1)(q-1)$;
- chooses e with $\gcd(e, \varphi(n)) = 1$;
- finds d with $ed = 1 \bmod \varphi(n)$;
- decrypts $c^d \bmod n = M$.

RSA: public and private numbers

A

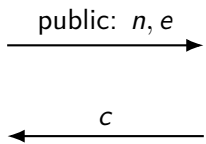
- chooses p, q secret;
- $n = pq$;
- $\varphi(n) = (p-1)(q-1)$;
- chooses e with $\gcd(e, \varphi(n)) = 1$;
- finds d with $ed = 1 \bmod \varphi(n)$;
- decrypts $c^d \bmod n = M$.



RSA: public and private numbers

A

- chooses p, q secret;
- $n = pq$;
- $\varphi(n) = (p-1)(q-1)$;
- chooses e with $\gcd(e, \varphi(n)) = 1$;
- finds d with $ed = 1 \bmod \varphi(n)$;
- decrypts $c^d \bmod n = M$.

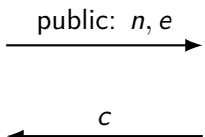


B

RSA: public and private numbers

A

- chooses p, q secret;
- $n = pq$;
- $\varphi(n) = (p-1)(q-1)$;
- chooses e with $\gcd(e, \varphi(n)) = 1$;
- finds d with $ed = 1 \bmod \varphi(n)$;
- decrypts $c^d \bmod n = M$.



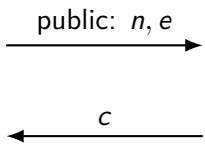
B

- has message M;

RSA: public and private numbers

A

- chooses p, q secret;
- $n = pq$;
- $\varphi(n) = (p-1)(q-1)$;
- chooses e with $\gcd(e, \varphi(n)) = 1$;
- finds d with $ed = 1 \bmod \varphi(n)$;
- decrypts $c^d \bmod n = M$.



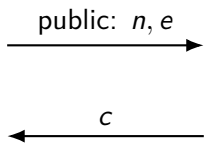
B

- has message M;
- computes $c = M^e \bmod n$;

RSA: public and private numbers

A

- chooses p, q secret;
- $n = pq$;
- $\varphi(n) = (p-1)(q-1)$;
- chooses e with $\gcd(e, \varphi(n)) = 1$;
- finds d with $ed = 1 \bmod \varphi(n)$;
- decrypts $c^d \bmod n = M$.



B

- has message M;
- computes $c = M^e \bmod n$;
- sends c to A.

RSA: public and private numbers continued

In practice, if $n = 4$, $p = 2$, $q = 2$, then

$$\varphi(n) = (2 - 1)(2 - 1) = 1.$$

RSA: public and private numbers continued

In practice, if $n = 4$, $p = 2$, $q = 2$, then

$$\varphi(n) = (2-1)(2-1) = 1.$$

If $n = 33$, then one can take $p = 3$, $q = 11$ and

$$\varphi(n) = 20.$$

RSA: public and private numbers continued

In practice, if $n = 4$, $p = 2$, $q = 2$, then

$$\varphi(n) = (2-1)(2-1) = 1.$$

If $n = 33$, then one can take $p = 3$, $q = 11$ and

$$\varphi(n) = 20.$$

Exercise: how long does it take to find p, q such that $n = 1387$?

RSA: public and private numbers continued

In practice, if $n = 4$, $p = 2$, $q = 2$, then

$$\varphi(n) = (2-1)(2-1) = 1.$$

If $n = 33$, then one can take $p = 3$, $q = 11$ and

$$\varphi(n) = 20.$$

Exercise: how long does it take to find p, q such that $n = 1387$?

Exercise: using quantum computers, can we speed up these factoring problems?

RSA: public and private numbers continued

In practice, if $n = 4$, $p = 2$, $q = 2$, then

$$\varphi(n) = (2-1)(2-1) = 1.$$

If $n = 33$, then one can take $p = 3$, $q = 11$ and

$$\varphi(n) = 20.$$

Exercise: how long does it take to find p, q such that $n = 1387$?

Exercise: using quantum computers, can we speed up these factoring problems?

THE END