# Quantum algorithms

## Lecture 1: Introduction and quantum circuits

Benoît Vermersch

March 1 2022

LPMMC Grenoble & IQOQI Innsbruck

# Outline

From classical to quantum computers

Lecture 1: Quantum circuits

- Single qubit states/gates
- Two qubit gates and universal quantum computing

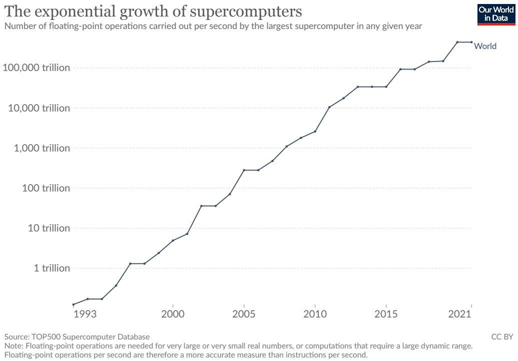
# Outline

From classical to quantum computers

Lecture 1: Quantum circuits

- Single qubit states/gates
- Two qubit gates and universal quantum computing

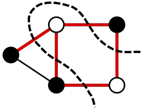
# The exponential growth of supercomputers




- Source Wikipedia
- Frontier (US) $10^{18}$ FLOPS - 600 million USD.
- Why do we need so much computation power?

## Some tough problems for classical computers

- Integer factorization: $N = ab$.
- Search algorithms: $f(x = w) = 1$, $f(x \neq w) = 0$. Find $w$.
- Optimization problems/Machine Learning: Given a cost function $f(x)$, find $x$ that maximizes $f(x)$



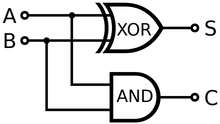
- Quantum problems: quantum chemistry, superconductivity, etc $\rightarrow$ solve large-scale Schrödinger equation

# Basic concepts of computer science

- A useful model for computers: circuits of logical gates acting on binary numbers.
- AND gate: $a' = ab \rightarrow$ Truth table:

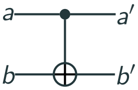
| a | b | a' |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

- OR gate: $a' = a + b \rightarrow$ Truth table: ...
- XOR gate: $a' = a \oplus b \rightarrow$ Truth table: ...
- Combining gates, we obtain *logical circuits*, eg Half-adder



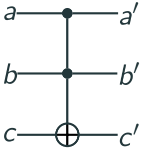
# Basic concepts of computer science

- Reversible circuits: $(a', b') = f(a, b)$, with $f$ invertible.
- Motivation: Irreversible processes increase entropy production during computation (Landauer's principle).
- Reversible XOR gate (known as CNOT in quantum computing)
  - $a' = a$, $b' = a \oplus b$
  - Remark: the second 'target' bit is flipped iff the first 'control' bit is activated ($a = 1$).
  - Useful notations for later



# Basic concepts of computer science

- Universality in reversible computing: Can I write using a finite set of gates an arbitary reversible circuit $f$, $(a'_1, \ldots, a'_n) = f(a_1, \ldots, a_n)$?
- The Toffoli gate is universal (see TD1)

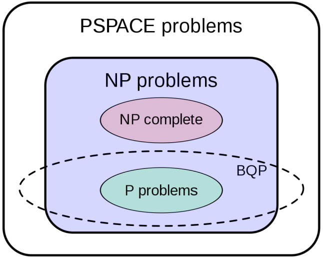


# Complexity classes

Complexity: scaling of resources to solve a decision problem (yes/no answer) with a classical computer (rigorously, for a deterministic Turing machine)

| | |
|---|---|
| P | Solved in polynomial time (ie number of operations) |
| NP | A yes answer verified in polynomial time |
| PSPACE | Solved with polynomial size (i.e number of constituents, bits) |
| EXPTIME | Solved in exponential time |
| NP-HARD | Every problem in NP can be transformed |
| | into this problem in polynomial time |
| NP-COMPLETE | A problem that is both NP and NP-HARD |

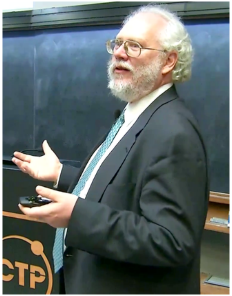
## Relations between complexity classes




- Conjecture P$\neq$NP
- BQP (Bounded error quantum polynomial time) is the class associated with quantum computers
- Conjecture P$\neq$BQP, ie quantum computers may be useful!

## Example: Integer factorization

- Factorization decision problem (F): can a given number be factorized?
- No polynomial time algorithm known: We do not know if F is in P
- Solutions can be checked efficiently: F is in NP
- Shor's algorithm (1995): F is in BQP




- Quantum computers may be able to tackle problems that are hard for classical computers!

# From classical to quantum computers in the quantum circuit model

Classical computers use classical bits

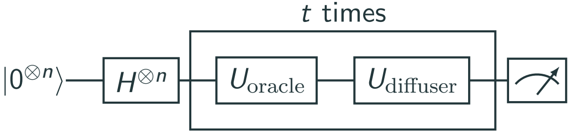
- One classical bit: 
  $\rightarrow$ state in 0 or 1
- $n$ classical bits:   . . . 
  $\rightarrow 2^n$ possibilities for the state
  $(00\ldots00,\ 00\ldots01,\text{etc})$

Quantum computers use qubits

- One qubit: 
  $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$
- $n$ qubits:   . . . 
  $|\psi\rangle = c_{0\ldots0}\,|0\ldots0\rangle + c_{0\ldots1}\,|0\ldots1\rangle + \ldots.$
  The quantum state can be simulatenously in all the $2^n$ classical states.

# Why we may expect quantum speedup with quantum paralelism

- Unstructured search on a space of $2^n$ bitstrings: We look for $x$, such that $f(x = x_1, \ldots, x_n) = 1$.
- Optimal classical algorithm: Random testing, with time complexity $O(2^n)$
- Optimal quantum algorithm: Grover's algorithm (Lecture 2)

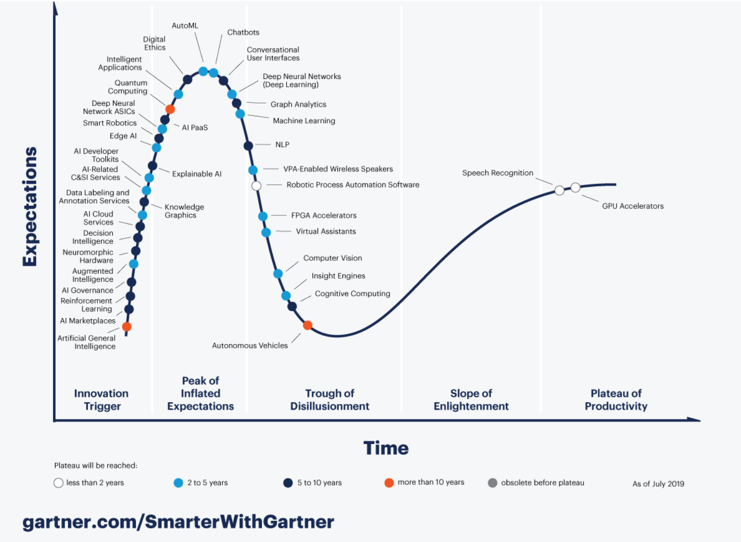



  - The first quantum gate creates a uniform quantum superposition of all bitstring states
  - Complexity $O(\sqrt{2^n})$: polynomial improvement (but still exponential scaling).

## Quantum Computing timeline

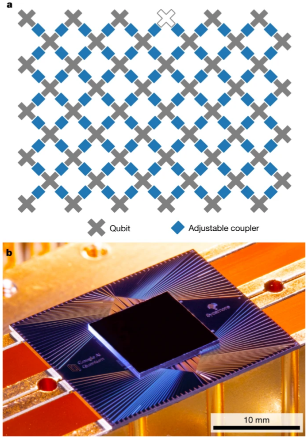
- 1980-1981: Concepts of quantum computers (Benioff-Manin-Feynman)
- 1985: Deutsch's universal quantum computer (Lecture 1)
- 1994: Shor's factoring algorithm (Lecture 3)
- 1995: Shor proposes quantum error correction (Lecture 4)
- 1995: First realization of a 2 qubit gate with trapped ions (Wineland)
- 1996: Grover's algorithm (Lecture 2)
- 2001: 15 is factorized with Shor's algorithm at IBM
- 2008: D-Wave Systems propose the first commercially available quantum computer (Lecture 5)
- 2019: Quantum supremacy claim by Google with 53 qubits (Lecture 6)
- 2021: IBM quantum eagle (127 qubits)

# The quantum ‘hype’



# Physical realizations: quantum hardware

- Superconducting qubits (Google, IBM, Rigetti, Grenoble, . . . )
- Trapped ions (Innsbruck, Duke university, Boulder NIST, IonQ, . . . ).
- Rydberg atoms (Palaiseau, Pasqal, Harvard, . . . )
- Electron spins (Delft, Microsoft, Grenoble, . . . )



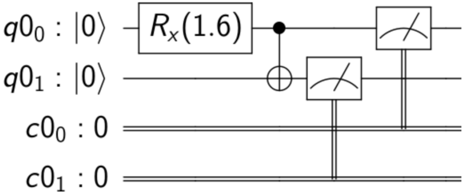
# IBMQ Practicals

- Organization: Julien Renard and BV.
- We will use via a cloud interface ‘small’ IBM quantum computers to illustrate the lectures.
- We will use the Qiskit Python library to parametrize simulate quantum circuits and interface with the quantum machines.

In [7]:

```
from qiskit import QuantumRegister, ClassicalRegister, QuantumCircui
from qiskit.tools.visualization import circuit_drawer
import numpy as np

qr = QuantumRegister(2)
cr = ClassicalRegister(2)
qp = QuantumCircuit(qr,cr)

qp.rx( np.pi/2,qr[0])
qp.cx(qr[0],qr[1])

qp.measure(qr,cr)

circuit_drawer(qp)
```

Out[7]:



# IBMQ Practicals: before the first practical

- Connect to Qiskit.org
- Install the Python Qiskit library on your laptop (eg via Anaconda)
- Test to import the library
- Create an IBMQ quantum experience account and get an API token

# Organization

- Lecture 1: Quantum circuits
- Lecture 2: Quantum algorithms (1)
- Lecture 3: Quantum algorithms (2)
- Lecture 4: Quantum error correction
- Lecture 5: Quantum simulation/quantum optimization
- Lecture 6: Bonus topics

# Resources

- Nielsen and Chuang, Quantum Computation and Quantum Information
- J. Preskill's quantum information lectures,
  `http://theory.caltech.edu/~preskill/`
- Lectures slides/Exercices/Schedule on Moodle Quantum Algorithms
- Groups/Schedule Moodle IBMQ

## Outline

From classical to quantum computers

Lecture 1: Quantum circuits

- Single qubit states/gates
- Two qubit gates and universal quantum computing

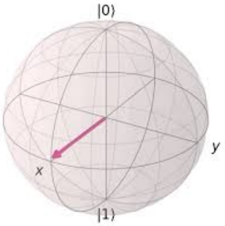
# Single qubit states

- For this course, the qubit can be thought as the elementary building block of a quantum computer.
- A qubit is a two-level quantum system

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \tag{1}$$

- Sometime it is instructive to represent a qubit as a vector on the Bloch sphere

$$|\psi\rangle = \cos(\frac{\theta}{2}) |0\rangle + \sin(\frac{\theta}{2}) e^{i\phi} |1\rangle \tag{2}$$



# Single qubit quantum circuit

- In single qubit quantum circuits, the qubit states evolves as a function of time, by successive applications of single qubit gates


- After the first gate, we obtain

$$|\psi_1\rangle = U_1 |\psi\rangle \,, \tag{3}$$

with $U_1 = e^{-iH_1 t}$ is a unitary $2 \times 2$ matrix (a rotation on the Bloch sphere)

- Can you write the final state of the circuit as a function of $U_1$, $U_2$, $U_3$?

## Important single qubit gates

- Note: It is important to get used to calculate the states of quantum circuits both using the matrix and the bra-ket notations

$X$-gate $\quad -\boxed{X}- \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad X = |0\rangle\langle 1| + |1\rangle\langle 0|$

$Z$-gate $\quad -\boxed{Z}- \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \quad Z = |0\rangle\langle 0| - |1\rangle\langle 1|$

$Y$-gate $\quad -\boxed{Y}- \quad Y = iXZ$

Hadamard-gate $\quad -\boxed{H}- \quad H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$

T-gate $\quad -\boxed{T}- \quad T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}$

and others..

- Can you generate the Hadamard gate with a circuit with $X$ and $Z$ gates? why?

# Multi-qubit circuit structure

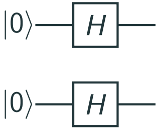
- A quantum circuit naturally extends to $n \geq 1$ qubits
- The wave-function is written in a tensor product space of dimension $2^n$

$$|\psi\rangle = \sum_{x_1=0}^{1} \sum_{x_2=0}^{1} \cdots \sum_{x_n=0}^{1} c_{x_1,x_2\ldots,x_n} |x_1\rangle \otimes |x_2\rangle \otimes \cdots \otimes |x_n\rangle = \begin{pmatrix} c_{0,0,\ldots 0,0} \\ c_{0,0\ldots 0,1} \\ \ldots \\ c_{1,1\ldots 1,1} \end{pmatrix} \tag{4}$$

- Equivalence between notations :
  $|0,1,1\rangle = |011\rangle = |0\rangle \otimes |1\rangle \otimes |1\rangle$, $|0\rangle \otimes |0\rangle = |0\rangle^{\otimes 2}$

# Multi-qubit circuit structure

- Let us calculate the state after the following two-qubit circuit?



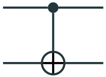

- Playing with bra-kets and tensor products

$$|\psi\rangle = (H \otimes H)(|0\rangle \otimes |0\rangle) = (H\,|0\rangle) \otimes (H\,|0\rangle) = \dots \tag{5}$$

- I can also first write $H$ in bra-ket notations then write $H \otimes H$ in bra-ket notations, or in matrix form, etc, but it's more tedious.
- Is this state entangled?

# Introducing two-qubit gates

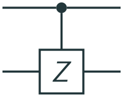
- Two qubit gates act non-trivially on two qubits
- Example CNOT gate



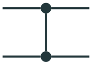
$$CNOT = |0\rangle\langle 0| \otimes 1 + |1\rangle\langle 1| \otimes X \tag{6}$$

Is this actually unitary?

- Example Controlled-Z gate



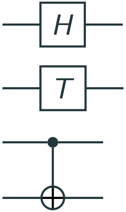

or



$$CZ = |0\rangle\langle 0| \otimes 1 + |1\rangle\langle 1| \otimes Z \tag{7}$$

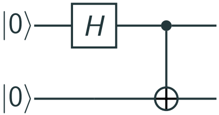
# The universal quantum computer

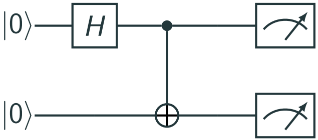
- Deutsch 1985: There exist universal set of gates that can be used to generate any quantum circuit $U$ acting on $n$ qubits.
- Note: The question of how many gates you need is non-trivial (quantum computational complexity)
- The following gate set is universal




- It is usually a good idea to use a larger gate set to simplify the circuit compilation.
- Write a circuit to create a Bell State, a three qubit GHZ state.

# The measurement

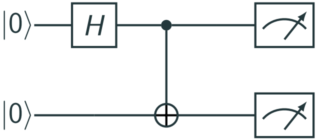
- Circuit to prepare the Bell state 

- Every quantum circuits ends up with a measurement 

  that produces random output $x_1, \ldots, x_n$ based on Born probabilities.

$$P(x_1, \ldots, x_n) = | \langle x_1, \ldots, x_n | \psi \rangle |^2 \tag{8}$$

# The measurement problem



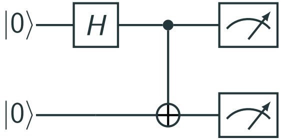

- I need to run the experiment $M$ time, accessing $m = 1, \ldots, M$ bitstrings $x_m = (x_{m,1}, \ldots, x_{m,n})$ to access meaningful information.

$$|\langle 00|\psi\rangle|^2 = \lim_{M\to\infty} \sum_{m=1}^{M} \frac{\delta_{x_m,(0,0)}}{M} \tag{9}$$

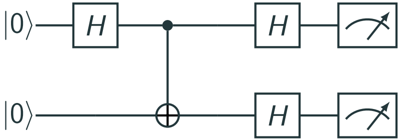
- The measurement problem is a crucial part in the design of quantum algorithms.

## Some common measurement circuits

- $Z$ Basis measurements: Gives access to Born probabilities and arbitrary expectation values involving only $Z$ operators.



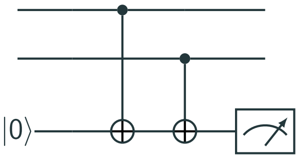

- One can also apply single qubit gates before the measurement, eg $X$ Basis measurements:




- The key identity is $Z = HXH$ (see $TD1$)

# Projection aspects and ancilla based measurements

- What happens if we do not measure entirely the system?
- Example: This measurement circuit takes a 2-qubit state $|\psi\rangle$, build a 3-qubit state $|\psi'\rangle$




and delivers one measurement outcome $\nu = 0, 1$.

- This is a (von Neumann) quantum measurement with projection operators $P_\nu = 1 \otimes 1 \otimes |\nu\rangle\langle\nu|$, $\nu = 0, 1$, which project the system into

$$P_\nu |\psi'\rangle \quad \text{with probability} \quad \langle\psi'|P_\nu|\psi'\rangle \tag{10}$$

- This type of measurements will be essential for error correction (Lecture 4).