

# Workshop on Quantum Computation using IBM Q, Part I

# Salvador E. Venegas-Andraca

Tecnológico de Monterrey, Escuela de Ingeniería y Ciencias



- 1 Thanks
- 2 Our goal
- 3 Who we are - QIP Group at Tecnológico de Monterrey
- 4 Quantum Computation -Definition and RDI ecosystem
- 5 Motivation and initial concepts - Quantum algorithms
- 6 A concise intro to Theoretical Computer Science
- 7 Mathematical preliminaries for Quantum Computation
- 8 Postulates of Quantum Mechanics
- 9 Our First Quantum Circuit!
- 10 A deeper look into quantum algorithms



Tecnológico  
de Monterrey



# My warmest thanks to (1/5)

**Our sponsors - Thank you very much!**



# My warmest thanks to (2/5)

**Our sponsors - Thank you very much!**



◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡

# My warmest thanks to (4/5)

## Our sponsors - Thank you very much!



Access to the D-Wave II quantum annealer @ NASA Ames Center

# My warmest thanks to (5/5)

**My warmest thanks to you all for taking part in this initiative!**

# Table of Contents

- 1 Thanks
- 2 **Our goal**
- 3 Who we are - QIP Group at Tecnológico de Monterrey
- 4 Quantum Computation -Definition and RDI ecosystem
- 5 Motivation and initial concepts - Quantum algorithms
- 6 A concise intro to Theoretical Computer Science
- 7 Mathematical preliminaries for Quantum Computation
- 8 Postulates of Quantum Mechanics
- 9 Our First Quantum Circuit!
- 10 A deeper look into quantum algorithms



**To contribute towards making quantum computing**

- i) A solid research field in Mexico and all Latin America**
- ii) An engine of regional economic development**

# Table of Contents

- 1 Thanks
- 2 Our goal
- 3 Who we are - QIP Group at Tecnológico de Monterrey**
- 4 Quantum Computation -Definition and RDI ecosystem
- 5 Motivation and initial concepts - Quantum algorithms
- 6 A concise intro to Theoretical Computer Science
- 7 Mathematical preliminaries for Quantum Computation
- 8 Postulates of Quantum Mechanics
- 9 Our First Quantum Circuit!
- 10 A deeper look into quantum algorithms

# Who we are

## **QIP Group at Tecnológico de Monterrey (1/2)**

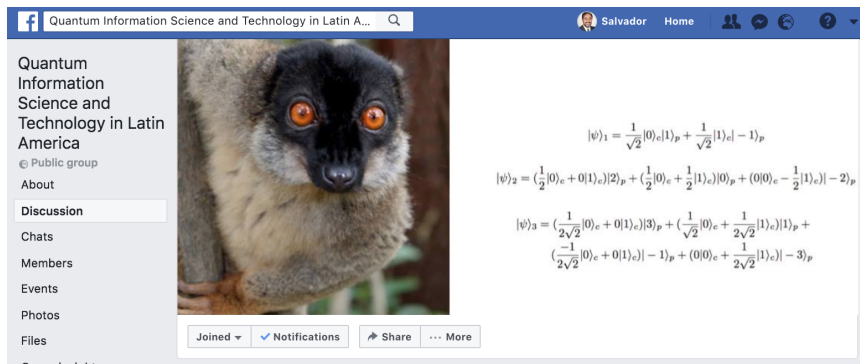
- We founded the field of quantum computation in Mexico.
- Quantum algorithms.
- Quantum Walks.
- Quantum annealing.
- Emerging fields of quantum computation, e.g. Quantum Image Processing and Quantum Machine Learning.
- Classical/digital simulation of quantum algorithms .
- Novel methods for training quantum scientists and quantum engineers.

# Who we are

## QIP Group at Tecnológico de Monterrey (2/2)

- Marco Lanzagorta. NRL, US Navy, USA. Quantum radar, quantum walks, quantum image processing, quantum annealing algorithms.
- Catherine McGeoch. D-Wave Systems, Canada. Quantum annealing algorithms.
- William de la Cruz de los Santos. UAEM. Quantum annealing algorithms, quantum computer vision.
- Mauricio García. IBM. Simulating and running quantum algorithms on IBM Q platforms.
- Enrique Hernández-Lemus. INMEGEN. Quantum algorithms for De novo Genome Sequence Assembly.
- Fei Yan, Changchun University, China and Ahmed Abdel-Latif, Menoufia University, Egypt. Quantum image processing.

# Who we are



The screenshot shows a Facebook group page. The header includes the Facebook logo, the group name "Quantum Information Science and Technology in Latin America", a search bar, and a profile picture of a man named Salvador. The left sidebar contains navigation links: "Quantum Information Science and Technology in Latin America", "Public group", "About", "Discussion" (highlighted), "Chats", "Members", "Events", "Photos", "Files", and "Group insights". The main content area features a large photo of a lemur with orange eyes. Below the photo are buttons for "Joined", "Notifications", "Share", and "More". To the right of the photo, three quantum state equations are displayed:

$$|\psi\rangle_1 = \frac{1}{\sqrt{2}}|0\rangle_c|1\rangle_p + \frac{1}{\sqrt{2}}|1\rangle_c|-1\rangle_p$$

$$|\psi\rangle_2 = (\frac{1}{2}|0\rangle_c + 0|1\rangle_c)|2\rangle_p + (\frac{1}{2}|0\rangle_c + \frac{1}{2}|1\rangle_c)|0\rangle_p + (0|0\rangle_c - \frac{1}{2}|1\rangle_c)|-2\rangle_p$$

$$|\psi\rangle_3 = (\frac{1}{2\sqrt{2}}|0\rangle_c + 0|1\rangle_c)|3\rangle_p + (\frac{1}{\sqrt{2}}|0\rangle_c + \frac{1}{2\sqrt{2}}|1\rangle_c)|1\rangle_p + (\frac{-1}{2\sqrt{2}}|0\rangle_c + 0|1\rangle_c)|-1\rangle_p + (0|0\rangle_c + \frac{1}{2\sqrt{2}}|1\rangle_c)|-3\rangle_p$$

Join our FB Group!

<https://www.facebook.com/groups/152996132018274/>

- 1 Thanks
- 2 Our goal
- 3 Who we are - QIP Group at Tecnológico de Monterrey
- 4 **Quantum Computation -Definition and RDI ecosystem**
- 5 Motivation and initial concepts - Quantum algorithms
- 6 A concise intro to Theoretical Computer Science
- 7 Mathematical preliminaries for Quantum Computation
- 8 Postulates of Quantum Mechanics
- 9 Our First Quantum Circuit!
- 10 A deeper look into quantum algorithms

# What is quantum computation?

Quantum computation, one of the most recent joint ventures between physics and computer science, may be defined as follows:

Quantum Computation is a multidisciplinary field focused on the development of **computers** and **algorithms**, *i.e.* **hardware** and **software**, based on the quantum mechanical properties of Nature.

# QC, a multidisciplinary and pervasive field

- Quantum computation is transitioning from an emerging branch of science into a mature research field. A growing number of quantum scientists & engineers are devoting their efforts to identifying and developing cross-fertilising initiatives in fields such as:
  - Artificial intelligence
  - Machine learning
  - Computational geometry
  - Image processing



# Many new players and interests

- Quantum computation is *also* an emerging market of advanced technology that is attracting:
  - **Investors.** *Breakoff Capital* ([www.breakoffcapital.com](http://www.breakoffcapital.com)), *Quantum Valley Investments* (<http://quantumvalleyinvestments.com/>).
  - **Global Companies.** *Microsoft* (<https://www.stationq.com/>), *Google* (<https://plus.google.com/+QuantumAILab>), *IBM* (<http://www.research.ibm.com/quantum/>)
  - **Entrepreneurs.** *1Qbit* (<http://1qbit.com/>), *Rigetti* (<http://rigetti.com/>), *D-Wave* (<http://www.dwavesys.com/>).
  - **Government.** *EU* (<https://ec.europa.eu/digital-single-market/en/news/european-commission-will-launch-eu1-billion-quantum-technologies-flagship>), *UK* (<http://uknqt.epsrc.ac.uk/>), *USA* (<https://www.whitehouse.gov/blog/2016/07/26/realizing-potential-quantum-information-science-and-advancing-high-performance>).

# Timeline (Theory)

## *Timeline (Theory)*



Feynman  
QM computers



Deutsch  
Universal QTM



Shor  
Integer factorization



Grover  
Search an unsorted DB



Nishimori  
Quantum Annealing



DiVincenzo  
Criteria to construct  
a QC



Aharonov  
Quantum Walks



Farhi  
Quantum Adiabatic  
Computation

## Companies (1/2)

### Companies (global)



Quantum Annealers (Optimization problems).  
SaS and direct selling.  
Classical simulation of quantum annealers

STATION



Microsoft

Topological quantum computation  
Classical simulators of quantum algorithms  
(Visual Studio-QC)



16, 25, 50 qubit universal computer. SaS.  
Software platform for classical simulation of quantum - Quantum simulation of quantum systems  
algorithms and development environment.



Google

-Universal quantum computer

## Companies (2/2)

### Companies (startups)



Quantum algorithms  
Machine learning



Quantum algorithms  
Quantum hardware



*Unconventional Computing*  
Quantum algorithms

# Table of Contents

- 1 Thanks
- 2 Our goal
- 3 Who we are - QIP Group at Tecnológico de Monterrey
- 4 Quantum Computation -Definition and RDI ecosystem
- 5 Motivation and initial concepts - Quantum algorithms**
- 6 A concise intro to Theoretical Computer Science
- 7 Mathematical preliminaries for Quantum Computation
- 8 Postulates of Quantum Mechanics
- 9 Our First Quantum Circuit!
- 10 A deeper look into quantum algorithms

# Quantum algorithms: fast and furious! (1/4)

A goal of paramount importance in quantum computation is to determine **how** to manipulate quantum mechanical systems in order to develop algorithms that are *faster* than their classical counterparts.

Here, *faster* has two meanings:

## Quantum algorithms: fast and furious! (2/4)

- Computational speed-up as a heritage of physical properties of quantum systems.

Example: quantum parallelism, i.e. quantum interference as a computational resource.

$$|0\rangle^{\otimes n} |0\rangle \xrightarrow{\hat{H}^{\otimes n} \hat{f}} \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle |0\rangle \xrightarrow{\hat{U}_f} \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle |f(i)\rangle$$

## Quantum algorithms: fast and furious! (3/4)

- Development of algorithmic strategies based on quantum mechanical properties.

Example in classical computer science: for solving the TSP we may use a brute force algorithm  $O(n!)$  or the Bellman–Held–Karp algorithm  $O(n^2 2^n)$ .

Some preliminary results in quantum computation but much is yet to be done!



# Quantum algorithms: fast and furious! (4/4)

Building quantum algorithms is a most challenging task because:

- a) First of all, we need a **complex interesting problem** (e.g., NP – complete, NP – hard) .
- b) Quantum mechanics is a counterintuitive theory and intuition plays a key role in algorithm design.
- c) A competitive quantum algorithm must not only produce the solutions it is expected to, **it also has to be more efficient, at least for some input values**, than any classical algorithm (**at least more efficient than existing classical algorithms**).

## Easy vs Difficult Algorithms

The graph illustrates the relative growth rates of different mathematical functions. The x-axis is labeled with values 5, 10, 15, 20, 25, 30, 35, and 40. The y-axis is labeled with values 1.E+00, 1.E+10, 1.E+20, 1.E+30, and 1.E+40. The functions are labeled as follows:

- $n!$  (Cyan line)
- $2^n$  (Purple line)
- $n^3$  (Green line)
- $n^2$  (Red line)
- $n$  (Blue line)

The graph shows that factorial growth ( $n!$ ) is the fastest, followed by exponential growth ( $2^n$ ), and then polynomial growth ( $n^3$ ,  $n^2$ , and  $n$ ).



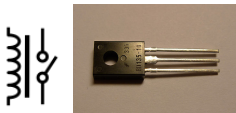
In order to motivate our discussion, let us briefly introduce the basic units of information in classical and quantum computation.

## Bits and Qubits - a quick introduction (2/6)

In classical computation, information is stored and manipulated in the form of **bits**.

The mathematical structure of a classical bit is rather simple: it suffices to define two logical values, traditionally labelled as  $\{0, 1\}$  (i.e. a classical bit lives in a scalar space) and to relate those values to two different and mutually exclusive outcomes of a classical measurement.

Examples:



A relay and a transistor  
(Source: Wikimedia Commons)

## Bits and Qubits - a quick introduction (3/6)

In quantum computation, information is stored, manipulated and measured in the form of **qubits (= quantum bit)**.

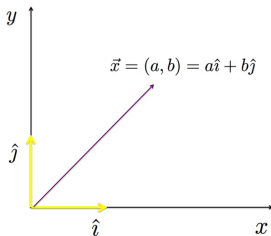
A qubit is a vector that represents the state of two-state quantum-mechanical physical entity.

# Bits and Qubits - a quick introduction (4/6)

Remember:

- $\mathbb{R}^2$ , the two-dimensional space studied in elementary school. Elements of  $\mathbb{R}^2$ , denoted by  $\vec{x}$ , are vectors.
- Any vector  $\vec{x} = (a, b)$  can be written as a linear combination of a basis, e.g.

$$\vec{x} = (a, b) = \vec{x} = (a, b) = a\hat{i} + b\hat{j}$$



# Bits and Qubits - a quick introduction (5/6)

In quantum computation, we use the following notation for denoting vectors:

$$\vec{x} = |x\rangle$$

So,

$$\vec{x} = a\hat{i} + b\hat{j} \Leftrightarrow |x\rangle = a|i\rangle + b|j\rangle$$

# Bits and Qubits - a quick introduction (6/6)

We shall shortly learn more about qubits but, before we do so, let us briefly focus on some key aspects of theoretical computer science.



## Python, QSKIT, Jupiter: installing and troubleshooting.

# Table of Contents

- 1 Thanks
- 2 Our goal
- 3 Who we are - QIP Group at Tecnológico de Monterrey
- 4 Quantum Computation -Definition and RDI ecosystem
- 5 Motivation and initial concepts - Quantum algorithms
- 6 A concise intro to Theoretical Computer Science**
- 7 Mathematical preliminaries for Quantum Computation
- 8 Postulates of Quantum Mechanics
- 9 Our First Quantum Circuit!
- 10 A deeper look into quantum algorithms

## What are the fundamental capabilities and limits of computers?

## Complexity theory. What makes some problems computationally hard and others easy?

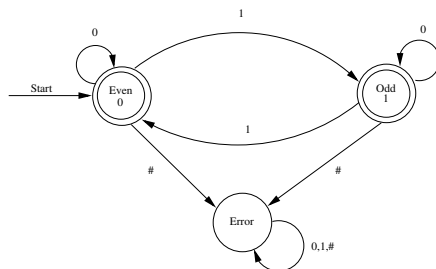
Automata are mathematical models of physical devices used to compute.

The definition of any automaton requires three components: states, language and transition rules.

## Automata (2/3)

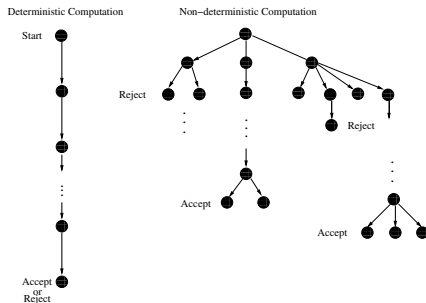
- States are abstractions of the different stages an automaton would go through while computing (e.g., *on* and *off* states of a switch).
- A language is a set of strings made by concatenating characters from a given alphabet. For example, one could define a language from alphabet  $\{a,b\}$ , consisting of all strings that begin with 'a' and end with 'b'. This language would include 'ab', 'aabb', and 'ababab', but not 'ba' or 'bbb'.
- Finally, transition rules describe the internal dynamics of an automaton (i.e., how and when an automaton changes its state).

# Automata (3/3) - An example of an automaton



A finite automaton for parity computation. Double-circled states are accept states and single-circled state is a reject state. Input strings for  $\mathcal{R}$  are any combination of characters taken from  $\Sigma = \{0, 1, \#\}$ . Since an empty set of characters has no 1s, we set the initial parity of a string as even (this is why the state 'Even' has an arrow labeled as 'Start').

# (Non)Deterministic Computation (1/3)



Deterministic automata follow a simple rule: given an input datum, every state of the computation is followed by only one state.



## (Non)Deterministic Computation (2/3)

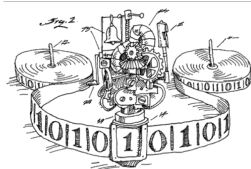
- Each and every automaton has deterministic and nondeterministic versions.
- Deterministic automata follow a simple rule : **given an input datum, every state of the computation is followed by only one state.**
- In contrast, when computing on a nondeterministic machine, **a state could be followed by many states**, depending on the input datum and the transition rules. Nondeterministic computation can be thought of as parallel computation with unlimited resources.

## (Non)Deterministic Computation (3/3)

- Resources are always limited, hence nondeterministic computation may seem to be an unreasonable model.
- However, there are two good reasons to use this model: i) to determine if a problem is computable even if limitless resources are available, and ii) to define a key set of computable problems: NP problems
- An **approximate** realization of non-deterministic computation: probabilistic algorithms.

# Turing Machines (1/3)

So far, Turing machines constitute the most accurate model of general purpose computers.



Turing Machines (drawing by Tom Dunne, American Scientist, March-April 2002)

The 'hardware' elements of a Deterministic Turing Machine (DTM) are a **limitless memory tape** (the tape is divided into squares or cells), and a **scanner which consists of a read-write head plus a finite state control system**. The scanner has two purposes: to read and write information on the cells of the tape as well as to control the state of the DTM.



# Turing Machines (3/3)

Three important properties of DTMs:

- 1) There is a Turing Machine, known as **The Universal Turing Machine** (UTM), that can simulate the behavior of any other Turing machine.
- 2) The UTM and the von Neumann computer (our silicon-based computers, like laptops) are equivalent in terms of computability power. In other words, any problem that can be solved using a von Neumann computer can also be solved using the UTM.
- 3) Although highly nontrivial, an arbitrary program written in any computer language can be translated into a series of elementary steps in a DTM.

# Algorithm Analysis

Algorithm analysis is divided into two parts:

- Algorithm Correctness.
- Algorithm resource consumption.

# Algorithm Correctness

An algorithm  $A$  is correct if:

- a) **[Design step]**  $A$  successfully does what it is designed to do. This step usually involves using solid mathematical statements (e.g., theorems).
- b) **[Implementation step]** Implementation of  $A$  is robust and free of bugs.

**Of course, easier said than done!**

Algorithm analysis also requires thinking about the amount of resources that an algorithm will consume during its execution. More precisely,

We need to understand/estimate how resource requirements will **scale** as input size increases.





# Resource consumption

**By Complexity Measure** we mean a definition of efficiency that is platform-independent, instance-independent and of predictive value with respect to increasing input sizes.

# Resource consumption

A measure based on specific implementation details like computer hardware instruction sets (e.g., Intel or AMD processors) or the primitives of a particular computer language (e.g., Python or Lisp) **would produce, for one and the same algorithm, different output values for different computer platforms.**

Consequently, we will formulate **abstract descriptions** to outline algorithm behavior.

The key concept here is, as you may expect, the notion of **elementary step**.

# Resource consumption

We shall analyze algorithm performance in two different scenarios:

- **Worst case complexity.** We are interested in estimating an **upper bound** on the **maximum** number of elementary steps that an algorithm must carry out. This is particularly important for pathological inputs.
- **Average case complexity.** We are interested in estimating the **expected** number of elementary steps that an algorithm must **typically** carry out. This is useful to estimate how an algorithm should perform 'in practice'.

In both cases, complexity is a function of input size  $|x|$ .

# Worst-case complexity

**Def. Worst-case complexity.** Let  $D_n$  be the set of inputs of size  $n$  for the problem under consideration and  $I \in D_n$ .

Also, let  $t(I)$  be the number of elementary steps performed by the algorithm on input  $I$ . We define the function  $W$  by

$$W = \max\{t(I) | I \in D_n\}$$

$W(n)$  is the maximum number of basic operations performed by the algorithm on *any* input of size  $n$ .

# Average-case complexity

**Def. Average-case complexity** Let  $D_n$  be the set of inputs of size  $n$  for the problem under consideration and  $I \in D_n$ .

Also, let  $t(I)$  be the number of elementary steps performed by the algorithm on input  $I$  and  $p(I)$  be the probability that input  $I$  occurs. We define the function  $A$  by

$$A = \sum_{I \in D_n} p(I)t(I)$$

## Example of worst-case and average-case complexity

**Problem.** Let  $L$  be an array containing  $n$  different entries. If  $x \in L$  return the index of  $x$  in  $L$  (that is, the array slot number in which  $x$  is contained). Return 0 otherwise (i.e., if  $x \notin L$ ).

**Elementary step.** Comparison of  $x$  with a list entry.

### Algorithm

```
index := 1  
while (index ≤  $n$  and  $L[\textit{index}] \neq x$ ) do  
  | index := index + 1  
end  
if index >  $n$  then  
  | index := 0  
end
```

## Example of worst-case and average-case complexity

**Worst case.** Worst cases occur when  $x$  appears only in the last position in the list and when  $x$  is not in the list at all.

In both cases, we need to compare  $x$  with all elements of  $L$ , i.e. execute  $n$  comparisons. Hence,

$$W(n) = n$$



## Example of worst-case and average-case complexity

**Average case.** We shall analyze two cases:

- a)  $x$  is an element of  $L$  .
- b)  $x$  may not be an element of  $L$ .

## Example of worst-case and average-case complexity

a)  $x \in L$ .

- Suppose that  $x$  is equally likely to be in any particular position.
- Let  $I_i, i \in \{1, 2, \dots, n\}$  represent the case where  $x$  appears in the  $i^{th}$  position of  $L$ .
- Let  $t(I_i)$  be the number of comparisons done (that is, the number of times the condition  $L[index] \neq x$  is tested).
- Note that, for  $i \in \{1, 2, \dots, n\}, t(I_i) = i$ .

Thus,

$$A(n) = \sum_{i=1}^n p(I_i) t(I_i) = \sum_{i=1}^n \frac{1}{n} i = \frac{1}{n} \sum_{i=1}^n i = \frac{1}{n} \frac{n(n+1)}{2} = \frac{(n+1)}{2}$$

So,

$$A(n) = \frac{(n+1)}{2}$$

## Example of worst-case and average-case complexity

b)  $x$  may not be an element of  $L$ .

- Let  $\alpha$  be the probability that  $x \in L$  and  $\beta = 1 - \alpha$  the probability that  $x \notin L$ .
- Suppose that if  $x \in L$  then  $x$  is equally likely to be in any particular position.
- Let  $I_i, i \in \{1, 2, \dots, n\}$  represent the case where  $x$  is in the  $i^{th}$  position of  $L$ .
- Let  $I_{n+1}$  represent the case in which  $x \notin L$ .
- Let  $t(I_i)$  be the number of comparisons done.
- Note that, for  $i \in \{1, 2, \dots, n\}, t(I_i) = i$  and  $t(I_{n+1}) = n$ .
- Finally, for  $i \in \{1, 2, \dots, n\}, p(I_i) = \frac{\alpha}{n}$  and  $p(I_{n+1}) = \beta$ .

## Example of worst-case and average-case complexity

Thus,

$$\begin{aligned}A(n) &= \sum_{i=1}^{n+1} p(I_i)t(I_i) \\&= \sum_{i=1}^n \frac{\alpha}{n}i + \beta I_{n+1} \\&= \frac{\alpha}{n} \sum_{i=1}^n i + \beta I_{n+1} \\&= \frac{\alpha}{n} \frac{n(n+1)}{2} + \beta I_{n+1} \\&= \frac{\alpha(n+1)}{2} + \beta I_{n+1}\end{aligned}$$

So,

$$A(n) = \frac{\alpha(n+1)}{2} + \beta I_{n+1}$$

# Intuitive Intro to Computational Complexity (1/10)

In complexity theory, for a problem  $\mathcal{P}$  that can be computed in a Turing machine, we are interested in answering the following question:

What is the minimum **amount of time or energy** / **number of elementary steps** that must be invested in order to solve  $\mathcal{P}$ ?

# Intuitive Intro to Computational Complexity (2/10)

## Wait! What do we mean by elementary step?

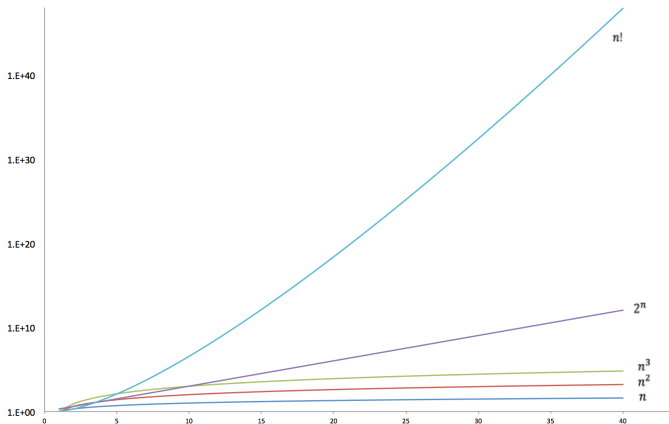
- Machine language
- Assembly language
- High level language
- Abstract algorithm (like computing the inner product between two vectors).

The key point: polynomial transformations from one level of algorithm description to another.

# Intuitive Intro to Computational Complexity (3/10)

## Easy vs Difficult Algorithms

(1 step/ns  $\Rightarrow 3.15 \times 10^{22}$  steps/one million years)



# Intuitive Intro to Computational Complexity (4/10)

Example of an easy problem: square matrix multiplication. Assume addition and multiplication as elementary steps.

$$\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ a_{21} & \cdots & a_{2n} \\ \vdots & & \vdots \\ a_{n1} & \cdots & a_{nn} \end{pmatrix} \begin{pmatrix} b_{11} & \cdots & b_{1n} \\ b_{21} & \cdots & b_{2n} \\ \vdots & & \vdots \\ b_{n1} & \cdots & b_{nn} \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^n a_{1i}b_{i1} & \cdots & \sum_{i=1}^n a_{1i}b_{in} \\ \sum_{i=1}^n a_{2i}b_{i1} & \cdots & \sum_{i=1}^n a_{2i}b_{in} \\ \vdots & & \vdots \\ \sum_{i=1}^n a_{ni}b_{i1} & \cdots & \sum_{i=1}^n a_{ni}b_{in} \end{pmatrix}$$

The textbook multiplication of two  $n \times n$  matrices requires  $n$  multiplications and  $n - 1$  additions per entry. Since the resulting matrix has  $n^2$  entries, then **the total amount of elementary steps required to multiply two  $n \times n$  matrices is**

$$(n + (n - 1))n^2 = 2n^3 - n^2$$



# Intuitive Intro to Computational Complexity (5/10)

For example, suppose that  $n = 1000$  and that a computer running the above-mentioned algorithm computes an elementary step in  $1 \mu s$  (which is a rather slow computer). So,

$$2n^3 - n^2|_{n=10^3} = 2 \times 10^9 - 10^6 = 1999 \times 10^6 \text{ elementary steps}$$

Now, if each elementary step takes  $1 \mu s$  then the total running time will be equal to

$$(1999 \times 10^6 \text{ elementary steps}) \left( 1 \times 10^{-6} \frac{s}{\text{elementary steps}} \right) = 1999s = 33.31 \text{ min}$$

which is perfectly fine.

In contrast, let us now analyze a key problem in theoretical computer science: the K-SAT problem.

# Intuitive Intro to Computational Complexity (7/10)

## K-SAT, a Fundamental NP-Complete Problem

Let  $B = \{x_1, x_2, \dots, x_n, \bar{x}_1, \bar{x}_2, \dots, \bar{x}_n\}$  be a set of Boolean variables. Also, let  $C_i$  be a disjunction of  $k$  elements of  $B$  and  $F$  be a conjunction of  $m$  clauses  $C_i$ .

Question: Is there an assignment of Boolean variables in  $F$  that satisfies all clauses simultaneously, i.e.  $F=1$ ?

For example:

# Intuitive Intro to Computational Complexity (8/10)

Suppose  $B = \{x_1, x_2, x_3, x_4, x_5, x_6, \bar{x}_1, \bar{x}_2, \bar{x}_3, \bar{x}_4, \bar{x}_5, \bar{x}_6\}$  and

$$\begin{aligned}
 P = & (\bar{x}_1 \vee \bar{x}_4 \vee \bar{x}_5) \wedge (\bar{x}_2 \vee \bar{x}_3 \vee \bar{x}_4) \wedge (x_1 \vee x_2 \vee \bar{x}_5) \wedge (x_3 \vee x_4 \vee x_5) \wedge \\
 & (x_4 \vee x_5 \vee \bar{x}_6) \wedge (\bar{x}_1 \vee \bar{x}_3 \vee \bar{x}_5) \wedge (x_1 \vee \bar{x}_2 \vee \bar{x}_5) \wedge (x_2 \vee \bar{x}_3 \vee \bar{x}_6) \wedge \\
 & (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_6) \wedge (x_3 \vee \bar{x}_5 \vee \bar{x}_6) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_4) \wedge (x_2 \vee x_3 \vee \bar{x}_4) \wedge \\
 & (x_2 \vee x_5 \vee \bar{x}_6) \wedge (x_2 \vee \bar{x}_3 \vee \bar{x}_5) \wedge (\bar{x}_2 \vee \bar{x}_3 \vee \bar{x}_4) \wedge (x_2 \vee x_3 \vee x_6) \wedge \\
 & (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee \bar{x}_4 \vee \bar{x}_5) \wedge (\bar{x}_3 \vee \bar{x}_4 \vee \bar{x}_6) \wedge (\bar{x}_4 \vee \bar{x}_5 \vee x_6) \wedge \\
 & (\bar{x}_2 \vee x_3 \vee \bar{x}_6) \wedge (x_2 \vee x_5 \vee x_6) \wedge (x_3 \vee x_5 \vee \bar{x}_6) \wedge (\bar{x}_1 \vee x_3 \vee \bar{x}_6) \wedge \\
 & (x_3 \vee \bar{x}_5 \vee x_6) \wedge (x_4 \vee x_5 \vee x_6) \wedge (x_1 \vee x_2 \vee \bar{x}_3)
 \end{aligned}$$

Finding the solutions (if any) of even a modest SAT instance can become difficult quite easily. In fact, only  $\{x_1 = 0, x_2 = 1, x_3 = 0, x_4 = 1, x_5 = 0, x_6 = 0\}$  satisfies  $P$ .

# Intuitive Intro to Computational Complexity (9/10)

Given an arbitrary instance  $F$ , we can always determine whether there is an assignment of Boolean variables such that  $F = 1$ .

To do so, we just have to substitute each and every possible value of  $x_i$ ,  $i \in \{1, \dots, n\}$  in  $F$ . However, this procedure would require us to produce  $2^n$  different assignments, i.e. an *exponential* number of different assignments.

# Intuitive Intro to Computational Complexity (10/10)

For example, suppose that  $n = 1000$  (remember,  $n$  is the number of binary variables) and that a computer running the above-mentioned brute force algorithm produces an assignment in  $1fs$ , i.e.  $10^{-15}s$  (a fast computer, indeed!)

Since  $2^{1000} \approx 1.0715086072 \times 10^{301}$  then the total amount of time to be invested in running our brute force algorithm would be about

$$1.0715086072 \times 10^{301} \times 10^{-15}s = 1.0715086072 \times 10^{286}s$$

i.e., about



$3.39 \times 10^{278}$  years!



# P, NP

Decision problems, a key concept in complexity theory, can be informally defined as yes/no questions.

The notion of decision problem is key to define the complexity classes known as P, NP and NP – complete. This is because decision problems can be put in correspondence with languages, being the latter a formal mathematical concept in computer science.

# P, NP

- **Class P**. For each decision problem  $\mathcal{A} \in P$  there is a DTM  $M$  and a polynomial  $p(n)$  such that processing input  $\omega$  of length  $n$  in  $M$  would be done in  $p(n)$  steps at most.



# P, NP

- **Class NP**. For each decision problem  $\mathcal{B} \in \text{NP}$  there is an NTM  $N$  and a polynomial  $p(n)$  such that processing input  $w$  of length  $n$  in  $N$  would be done in  $p(n)$  steps at most.

# P, NP

**Theorem.** If decision problem  $A$  is in NP then there is a polynomial  $p(n)$ , where  $n$  is the size of an arbitrary input data, such that  $A$  can be solved by a deterministic algorithm having time complexity  $O(2^{p(n)})$ .

# P, NP

**Equivalent definition for NP.** A problem  $\mathcal{B}$  is in NP if there is a polynomial time algorithm designed to test whether a proposed solution to  $\mathcal{B}$  is correct.

Note: worrying about how hard it might be to find the solution is not part of this definition 😊

So, if only we could guess the right solution, we could then quickly test it.

# P, NP

It is well known that  $P \subseteq NP$  and it is unknown whether  $NP \subseteq P$ .

Hence, a key research area in contemporary computer science is the study of decision problems in NP for which we only know inefficient algorithms, i.e. algorithms with time complexity functions that are not upper-bounded by polynomials.

# Example 1 - NP problems /1/2)

## Solving hard instances of the K-SAT problem

### K-SAT

Let  $B = \{x_1, x_2, \dots, x_n, \bar{x}_1, \bar{x}_2, \dots, \bar{x}_n\}$  be a set of Boolean variables. Also, let  $C_i$  be a disjunction of  $k$  elements of  $B$  and  $F$  be a conjunction of  $m$  clauses  $C_i$ .

Question: Is there an assignment of Boolean variables in  $F$  that satisfies all clauses simultaneously, i.e.  $F=1$ ?

---

## Example 1 - NP problems (2/2)

Suppose  $B = \{x_1, x_2, x_3, x_4, x_5, x_6, \bar{x}_1, \bar{x}_2, \bar{x}_3, \bar{x}_4, \bar{x}_5, \bar{x}_6\}$  and

$$\begin{aligned}
 P = & (\bar{x}_1 \vee \bar{x}_4 \vee \bar{x}_5) \wedge (\bar{x}_2 \vee \bar{x}_3 \vee \bar{x}_4) \wedge (x_1 \vee x_2 \vee \bar{x}_5) \wedge (x_3 \vee x_4 \vee x_5) \wedge \\
 & (x_4 \vee x_5 \vee \bar{x}_6) \wedge (\bar{x}_1 \vee \bar{x}_3 \vee \bar{x}_5) \wedge (x_1 \vee \bar{x}_2 \vee \bar{x}_5) \wedge (x_2 \vee \bar{x}_3 \vee \bar{x}_6) \wedge \\
 & (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_6) \wedge (x_3 \vee \bar{x}_5 \vee \bar{x}_6) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_4) \wedge (x_2 \vee x_3 \vee \bar{x}_4) \wedge \\
 & (x_2 \vee x_5 \vee \bar{x}_6) \wedge (x_2 \vee \bar{x}_3 \vee \bar{x}_5) \wedge (\bar{x}_2 \vee \bar{x}_3 \vee \bar{x}_4) \wedge (x_2 \vee x_3 \vee x_6) \wedge \\
 & (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee \bar{x}_4 \vee \bar{x}_5) \wedge (\bar{x}_3 \vee \bar{x}_4 \vee \bar{x}_6) \wedge (\bar{x}_4 \vee \bar{x}_5 \vee x_6) \wedge \\
 & (\bar{x}_2 \vee x_3 \vee \bar{x}_6) \wedge (x_2 \vee x_5 \vee x_6) \wedge (x_3 \vee x_5 \vee \bar{x}_6) \wedge (\bar{x}_1 \vee x_3 \vee \bar{x}_6) \wedge \\
 & (x_3 \vee \bar{x}_5 \vee x_6) \wedge (x_4 \vee x_5 \vee x_6) \wedge (x_1 \vee x_2 \vee \bar{x}_3)
 \end{aligned}$$

Finding the solutions (if any) of even a modest SAT instance can become difficult quite easily. In fact, only  $\{x_1 = 0, x_2 = 1, x_3 = 0, x_4 = 1, x_5 = 0, x_6 = 0\}$  satisfies  $P$ .

Notice that, for an binary string  $S$ , it would be possible to quickly determine whether  $S$  is a solution to an instance of the K-SAT problem.

## Example 2 - Optimisation and NP problems (1/2)

**Def. The Traveling Salesman Problem (optimization version)**

INSTANCE: A finite set  $C = \{c_1, c_2, \dots, c_m\}$  of “cities” and a “distance”  $d(c_i, c_j) \in \mathbb{N}$ , the set of natural numbers.

QUESTION: Which is the shortest “tour” of all the cities in  $C$ , that is, an ordering  $[c_{\Pi(1)}, c_{\Pi(2)}, \dots, c_{\Pi(m)}]$  of  $C$  such that

$$\left[ \sum_{i=1}^{m-1} d(c_{\Pi(i)}, c_{\Pi(i+1)}) \right] + d(c_{\Pi(m)}, c_{\Pi(1)})$$

is minimum?

Suppose we were given a tour  $T_i$ . Would it be possible to **quickly** check whether  $T_i$  is the shortest tour of all possible tours?

## Example 2 - Optimisation and NP problems (2/2)

**Def. The Traveling Salesman Problem (decision problem version)**

INSTANCE: A finite set  $C = \{c_1, c_2, \dots, c_m\}$  of “cities” and a “distance”  $d(c_i, c_j) \in \mathbb{N}$  and a bound  $B \in \mathbb{N}$ .

QUESTION: Is there a “tour” of all the cities in  $C$  having total length no more than  $B$ , that is, an ordering  $[c_{\Pi(1)}, c_{\Pi(2)}, \dots, c_{\Pi(m)}]$  of  $C$  such that  $[\sum_{i=1}^{m-1} d(c_{\Pi(i)}, c_{\Pi(i+1)})] + d(c_{\Pi(m)}, c_{\Pi(1)}) \leq B$ ?



## Introductions to Computational Complexity for Physicists (1/2)

### LIMITS OF COMPUTATION

Copyright (c) 2002 Institute of Electrical and Electronics Engineers. Reprinted, with permission, from *Computing in Science & Engineering*. This material is posted here with permission of the IEEE. Such permission of the IEEE does not in any way imply IEEE endorsement of any of the discussed products or services. Internal or personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution must be obtained from the IEEE by sending a blank email message to [info.pub.permissions@ieee.org](mailto:info.pub.permissions@ieee.org). By choosing to view this document, you agree to all provisions of the copyright laws protecting it.

## COMPUTATIONAL COMPLEXITY FOR PHYSICISTS

*The theory of computational complexity has some interesting links to physics, in particular to quantum computing and statistical mechanics. This article contains an informal introduction to this theory and its links to physics.*

S. Mertens. IEEE Journal on Computing in Science & Engineering  
vol. 4, no. 3, May/June 2002, pp 31-47

# Introductions to Computational Complexity for Physicists (2/2)

CONTEMPORARY PHYSICS, 2018  
<https://doi.org/10.1080/00107514.2018.1450720>



Taylor & Francis  
 Taylor & Francis Group



## A cross-disciplinary introduction to quantum annealing-based algorithms

Salvador E. Venegas-Andraca<sup>a</sup> , William Cruz-Santos<sup>b</sup>, Catherine McGeoch<sup>c</sup> and Marco Lanzagorta<sup>d</sup>

<sup>a</sup>Escuela de Ingeniería y Ciencias, Tecnológico de Monterrey, Monterrey, Mexico; <sup>b</sup>CU-UAEM Valle de Chalco, Estado de México, Mexico;

<sup>c</sup>D-Wave Systems, Burnaby, Canada; <sup>d</sup>US Naval Research Laboratory, Washington, DC, USA

### ABSTRACT

A central goal in quantum computing is the development of quantum hardware and quantum algorithms in order to analyse challenging scientific and engineering problems. Research in quantum computation involves contributions from both physics and computer science; hence this article presents a concise introduction to basic concepts from both fields that are used in annealing-based quantum computation, an alternative to the more familiar quantum gate model. We introduce some concepts from computer science required to define difficult computational problems and to realise the potential relevance of quantum algorithms to find novel solutions to those problems. We introduce the structure of quantum annealing-based algorithms as well as two examples of this kind of algorithms for solving instances of the max-SAT and Minimum Multicut problems. An overview of the quantum annealing systems manufactured by D-Wave Systems is also presented.

### ARTICLE HISTORY

Received 12 June 2017

Accepted 6 March 2018

### KEYWORDS

Quantum annealing;  
 quantum algorithms;  
 quantum computation

So far,

We have learned some basic notions of automata and complexity theory, certainly enough to realize the actual and potential relevance of quantum computation.

Let us now focus on the mathematics of quantum computation.



# Table of Contents

- 1 Thanks
- 2 Our goal
- 3 Who we are - QIP Group at Tecnológico de Monterrey
- 4 Quantum Computation -Definition and RDI ecosystem
- 5 Motivation and initial concepts - Quantum algorithms
- 6 A concise intro to Theoretical Computer Science
- 7 Mathematical preliminaries for Quantum Computation**
- 8 Postulates of Quantum Mechanics
- 9 Our First Quantum Circuit!
- 10 A deeper look into quantum algorithms

# Hilbert space

A **Hilbert space**  $\mathcal{H}$  is a (complete) complex inner-product vector space. An example of a Hilbert space is  $\mathbb{C}^n$ .

# Kets and Bras

The Bra-Ket notation, also known as Dirac Notation, is a standard representation to describe quantum states.

The Dirac notation is widely used in quantum mechanics and quantum computation.

Let us now formally define the notions of Ket and Bra.

# Kets and Bras

**Kets.** Let  $\mathcal{H}$  be a Hilbert space. A vector  $\psi \in \mathcal{H}$  is denoted by  $|\psi\rangle$  and it is referred to as a **ket**.

We can represent elements  $|\psi\rangle$  of  $\mathcal{H}$  as column vectors by choosing a basis for  $\mathcal{H}$ . For example, let  $\mathcal{H} = \mathbb{C}^2$  and let us choose the vector basis  $\{|0\rangle, |1\rangle\}$ , where

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \text{ and } |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Then, every element  $|\psi\rangle \in \mathcal{H}$  can be written as

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \alpha \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \alpha, \beta \in \mathbb{C}$$

## Example of kets

$|\psi\rangle = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \in \mathbb{C}^2$  may be written as

$$|\psi\rangle = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$



## Exercise - Kets

Let  $|+\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$  and  $|-\rangle = \frac{|0\rangle-|1\rangle}{\sqrt{2}}$ . Write  $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$  in terms of  $|+\rangle, |-\rangle$ .

## Answer to exercise - Kets

Let  $|+\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$  and  $|-\rangle = \frac{|0\rangle-|1\rangle}{\sqrt{2}}$ . Write  $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$  in terms of  $|+\rangle, |-\rangle$ .

$$|+\rangle + |-\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}} + \frac{|0\rangle-|1\rangle}{\sqrt{2}} = \frac{2|0\rangle}{\sqrt{2}} \Rightarrow |0\rangle = \frac{|+\rangle+|-\rangle}{\sqrt{2}}$$

Similarly,

$$|+\rangle - |-\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}} - \left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right) = \frac{2|1\rangle}{\sqrt{2}} \Rightarrow |1\rangle = \frac{|+\rangle-|-\rangle}{\sqrt{2}}$$

Hence,

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \alpha\frac{|+\rangle+|-\rangle}{\sqrt{2}} + \beta\frac{|+\rangle-|-\rangle}{\sqrt{2}} = \frac{\alpha+\beta}{\sqrt{2}}|+\rangle + \frac{\alpha-\beta}{\sqrt{2}}|-\rangle$$

Therefore,

$$|\psi\rangle = \frac{\alpha+\beta}{\sqrt{2}}|+\rangle + \frac{\alpha-\beta}{\sqrt{2}}|-\rangle$$

# Bras

**Bras.** Formally speaking, bras are functionals (i.e. functions of vector spaces into corresponding fields) and in practice, they can be thought of as row vectors:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \text{ if and only if } \langle\psi| = \alpha^*\langle 0| + \beta^*\langle 1|$$

# Bras

For example, let

$$|\psi\rangle = \frac{i}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{i}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{i}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

The corresponding bra  $\langle\psi|$  is

$$\langle\psi| = \frac{-i}{\sqrt{2}}(1, 0) + \frac{1}{\sqrt{2}}(0, 1) = \left(\frac{-i}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right) = \frac{-i}{\sqrt{2}}\langle 0| + \frac{1}{\sqrt{2}}\langle 1|$$

# Exercises - Bras

1. Compute  $\langle + |$  and  $\langle - |$
2. Let  $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$  where  $||\alpha||^2 + ||\beta||^2 = 1$ . Does it follow that  $\langle\psi| = \alpha^*\langle 0| + \beta^*\langle 1|$  *where*  $||\alpha^*||^2 + ||\beta^*||^2 = 1$ ?

# Answers to exercises - Bras

1. Compute  $\langle + |$  and  $\langle - |$

Answer:  $\langle + | = \frac{\langle 0 | + \langle 1 |}{\sqrt{2}}$  and  $\langle - | = \frac{\langle 0 | - \langle 1 |}{\sqrt{2}}$

## Answers to exercises - Bras

2. Let  $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$  where  $||\alpha||^2 + ||\beta||^2 = 1$ . Does it follow that  $\langle\psi| = \alpha^*\langle 0| + \beta^*\langle 1|$  where  $||\alpha^*||^2 + ||\beta^*||^2 = 1$ ?

Answer:

a)  $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \Rightarrow \langle\psi| = \alpha^*\langle 0| + \beta^*\langle 1|$ .

b) Since  $\alpha, \beta \in \mathbb{C}$  then let us write  $\alpha = a + bi$  and  $\beta = c + di$ .

c) Also,  $\alpha^* = a - bi$  and  $\beta^* = c - di$

d) Furthermore,  $||\alpha||^2 = a^2 + b^2$  and  $||\beta||^2 = c^2 + d^2 \Rightarrow$

$$||\alpha||^2 + ||\beta||^2 = a^2 + b^2 + c^2 + d^2 = 1$$

e) Finally, please note that  $||\alpha^*||^2 = a^2 + b^2$  and  $||\beta^*||^2 = c^2 + d^2 \Rightarrow$

$$||\alpha^*||^2 + ||\beta^*||^2 = a^2 + b^2 + c^2 + d^2 = 1$$

So, the answer is **Yes, it does**.

# Summary of Kets and Bras

Thus, if  $\mathcal{H}$  is an  $n$ -dimensional Hilbert space then

- A ket  $|\psi\rangle \in \mathcal{H}$  can be represented as an  $n$ -dimensional column vector.
- Its corresponding bra  $\langle\psi| \in \mathcal{H}^*$  can be seen as an  $n$ -dimensional row vector

$|\psi\rangle \leftrightarrow \langle\psi|$  **corresponds to transposition and conjugation.**



# Inner product

We can use the Dirac notation to make calculations. For example,  $\langle\phi|\psi\rangle$  is the usual row-column matrix operator that computes the inner product in finite dimensional vector spaces.

For instance, let us take the representations of  $|0\rangle$  and  $|1\rangle$  given in previous slides

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \text{ and } |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Note that  $|0\rangle \perp |1\rangle$  as well as the fact that both vectors have unitary norm. Consequently, the inner product of  $|0\rangle$  and  $|1\rangle$  must be zero and the inner product of each vector with itself must be equal to one:

# Inner product

$$\langle \mathbf{0} | \mathbf{1} \rangle = (1, 0) \begin{pmatrix} 0 \\ 1 \end{pmatrix} = (1 \times 0 + 0 \times 1) = \mathbf{0} = (0, 1) \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \langle \mathbf{1} | \mathbf{0} \rangle$$

Moreover

$$\langle \mathbf{0} | \mathbf{0} \rangle = (1, 0) \begin{pmatrix} 1 \\ 0 \end{pmatrix} = (1 \times 1 + 0 \times 0) = \mathbf{1} = (0, 1) \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \langle \mathbf{1} | \mathbf{1} \rangle$$

# Exercises inner product

- 1 Let  $|\psi\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$  and  $|\phi\rangle = \frac{\sqrt{3}}{2}|0\rangle + \frac{1}{2}|1\rangle$ . Compute  $\langle\psi|\phi\rangle$  and  $\langle\phi|\psi\rangle$
- 2 Let  $|\psi\rangle = \frac{i}{\sqrt{2}}|0\rangle + \frac{i}{\sqrt{2}}|1\rangle$  and  $|\phi\rangle = \frac{3}{4}|0\rangle + \frac{\sqrt{7}i}{4}|1\rangle$ . Compute  $\langle\psi|\phi\rangle$  and  $\langle\phi|\psi\rangle$ .

# Answers to exercises inner product

$$1. |\psi\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \text{ and } |\phi\rangle = \frac{\sqrt{3}}{2}|0\rangle + \frac{1}{2}|1\rangle$$

$$\langle\psi|\phi\rangle = \left(\frac{1}{\sqrt{2}}\langle 0| + \frac{1}{\sqrt{2}}\langle 1|\right)\left(\frac{\sqrt{3}}{2}|0\rangle + \frac{1}{2}|1\rangle\right)$$

$$= \frac{1}{\sqrt{2}} \times \frac{\sqrt{3}}{2} \langle 0|0\rangle + \frac{1}{\sqrt{2}} \times \frac{1}{2} \langle 0|1\rangle + \frac{1}{\sqrt{2}} \times \frac{\sqrt{3}}{2} \langle 1|0\rangle + \frac{1}{\sqrt{2}} \times \frac{1}{2} \langle 1|1\rangle$$

$$= \frac{1}{\sqrt{2}} \times \frac{\sqrt{3}}{2} \times 1 + \frac{1}{\sqrt{2}} \times \frac{1}{2} \times 0 + \frac{1}{\sqrt{2}} \times \frac{\sqrt{3}}{2} \times 0 + \frac{1}{\sqrt{2}} \times \frac{1}{2} \times 1$$

$$= \frac{\sqrt{3}+1}{2\sqrt{2}}$$

# Answers to exercises inner product

$$1. |\psi\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \text{ and } |\phi\rangle = \frac{\sqrt{3}}{2}|0\rangle + \frac{1}{2}|1\rangle$$

$$\langle\phi|\psi\rangle = \left(\frac{\sqrt{3}}{2}\langle 0| + \frac{1}{2}\langle 1|\right)\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right)$$

$$= \frac{\sqrt{3}}{2} \times \frac{1}{\sqrt{2}} \langle 0|0\rangle + \frac{\sqrt{3}}{2} \times \frac{1}{\sqrt{2}} \langle 0|1\rangle + \frac{1}{2} \times \frac{1}{\sqrt{2}} \langle 1|0\rangle + \frac{1}{2} \times \frac{1}{\sqrt{2}} \langle 1|1\rangle$$

$$= \frac{\sqrt{3}}{2} \times \frac{1}{\sqrt{2}} \times 1 + \frac{\sqrt{3}}{2} \times \frac{1}{\sqrt{2}} \times 0 + \frac{1}{2} \times \frac{1}{\sqrt{2}} \times 0 + \frac{1}{2} \times \frac{1}{\sqrt{2}} \times 1$$

$$= \frac{\sqrt{3}+1}{2\sqrt{2}}$$

# Answers to exercises inner product

$$2. |\psi\rangle = \frac{i}{\sqrt{2}}|0\rangle + \frac{i}{\sqrt{2}}|1\rangle \text{ and } |\phi\rangle = \frac{3}{4}|0\rangle + \frac{\sqrt{7}i}{4}|1\rangle.$$

$$\begin{aligned} \langle\psi|\phi\rangle &= \left(\frac{-i}{\sqrt{2}}\langle 0| + \frac{-i}{\sqrt{2}}\langle 1|\right)\left(\frac{3}{4}|0\rangle + \frac{\sqrt{7}i}{4}|1\rangle\right) \\ &= \frac{-i}{\sqrt{2}} \times \frac{3}{4} \langle 0|0\rangle + \frac{-i}{\sqrt{2}} \times \frac{\sqrt{7}i}{4} \langle 0|1\rangle + \frac{-i}{\sqrt{2}} \times \frac{3}{4} \langle 1|0\rangle + \frac{-i}{\sqrt{2}} \times \frac{\sqrt{7}i}{4} \langle 1|1\rangle \\ &= \frac{-i}{\sqrt{2}} \times \frac{3}{4} \times 1 + \frac{-i}{\sqrt{2}} \times \frac{\sqrt{7}i}{4} \times 0 + \frac{-i}{\sqrt{2}} \times \frac{3}{4} \times 0 + \frac{-i}{\sqrt{2}} \times \frac{\sqrt{7}i}{4} \times 1 \\ &= \frac{\sqrt{7}}{4\sqrt{2}} - \frac{3}{4\sqrt{2}}i \end{aligned}$$

# Answers to exercises inner product

$$2. |\psi\rangle = \frac{i}{\sqrt{2}}|0\rangle + \frac{i}{\sqrt{2}}|1\rangle \text{ and } |\phi\rangle = \frac{3}{4}|0\rangle + \frac{\sqrt{7}i}{4}|1\rangle.$$

$$\begin{aligned} \langle\phi|\psi\rangle &= \left(\frac{3}{4}\langle 0| + \frac{-\sqrt{7}i}{4}\langle 1|\right)\left(\frac{i}{\sqrt{2}}|0\rangle + \frac{i}{\sqrt{2}}|1\rangle\right) \\ &= \frac{3}{4} \times \frac{i}{\sqrt{2}}\langle 0|0\rangle + \frac{3}{4} \times \frac{i}{\sqrt{2}}\langle 0|1\rangle + \frac{-\sqrt{7}i}{4} \times \frac{i}{\sqrt{2}}\langle 1|0\rangle + \frac{-\sqrt{7}i}{4} \times \frac{i}{\sqrt{2}}\langle 1|1\rangle \\ &= \frac{3}{4} \times \frac{i}{\sqrt{2}} \times 1 + \frac{3}{4} \times \frac{i}{\sqrt{2}} \times 0 + \frac{-\sqrt{7}i}{4} \times \frac{i}{\sqrt{2}} \times 0 + \frac{-\sqrt{7}i}{4} \times \frac{i}{\sqrt{2}} \times 1 \\ &= \frac{\sqrt{7}}{4\sqrt{2}} + \frac{3}{4\sqrt{2}}i \end{aligned}$$

# Outer product

We can also use the Dirac notation to compute vectors. Let  $|\psi\rangle, |a\rangle \in \mathcal{H}_1$  and  $|\phi\rangle \in \mathcal{H}_2$  then the *outer product* is the linear operator from  $\mathcal{H}_1$  to  $\mathcal{H}_2$  defined by

$$(|\phi\rangle\langle\psi|)|a\rangle \equiv (\langle\psi|a\rangle)|\phi\rangle$$

As it may be expected, the summation of outer products is also a linear operator.



## Example - Outer product

For example, let us define the Hadamard operator

$$\hat{H} = \frac{1}{\sqrt{2}}(|0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| - |1\rangle\langle 1|)$$

The action of  $\hat{H}$  on ket  $|0\rangle$  is given by

$$\begin{aligned}\hat{H}|0\rangle &= \left(\frac{1}{\sqrt{2}}|0\rangle\langle 0| + \frac{1}{\sqrt{2}}|0\rangle\langle 1| + \frac{1}{\sqrt{2}}|1\rangle\langle 0| - \frac{1}{\sqrt{2}}|1\rangle\langle 1|\right)|0\rangle \\ &= \frac{\langle 0|0\rangle}{\sqrt{2}}|0\rangle + \frac{\langle 1|0\rangle}{\sqrt{2}}|0\rangle + \frac{\langle 0|0\rangle}{\sqrt{2}}|1\rangle - \frac{\langle 1|0\rangle}{\sqrt{2}}|1\rangle \\ &= \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\end{aligned}$$

# Exercise - Outer product

Let  $\hat{\sigma}_y = i|0\rangle\langle 1| - i|1\rangle\langle 0|$  and  $|\psi\rangle = \frac{\sqrt{3}}{2}|0\rangle + \frac{i}{2}|1\rangle$ . Compute  $\hat{\sigma}_y|\psi\rangle$ .

# Answer to exercise - Outer product

Let  $\hat{\sigma}_y = i|0\rangle\langle 1| - i|1\rangle\langle 0|$  and  $|\psi\rangle = \frac{\sqrt{3}}{2}|0\rangle + \frac{i}{2}|1\rangle$ . Compute  $\hat{\sigma}_y|\psi\rangle$ .

$$\begin{aligned}
 \hat{\sigma}_y|\psi\rangle &= (i|0\rangle\langle 1| - i|1\rangle\langle 0|)(\frac{\sqrt{3}}{2}|0\rangle + \frac{i}{2}|1\rangle) \\
 &= \frac{\sqrt{3}i\langle 1|0\rangle}{2}|0\rangle + \frac{i^2\langle 1|1\rangle}{2}|0\rangle - \frac{\sqrt{3}i\langle 0|0\rangle}{2}|1\rangle - \frac{i^2\langle 0|1\rangle}{2}|1\rangle \\
 &= -\frac{1}{2}|0\rangle - \frac{\sqrt{3}i}{2}|1\rangle
 \end{aligned}$$

- 1 Thanks
- 2 Our goal
- 3 Who we are - QIP Group at Tecnológico de Monterrey
- 4 Quantum Computation -Definition and RDI ecosystem
- 5 Motivation and initial concepts - Quantum algorithms
- 6 A concise intro to Theoretical Computer Science
- 7 Mathematical preliminaries for Quantum Computation
- 8 Postulates of Quantum Mechanics**
- 9 Our First Quantum Circuit!
- 10 A deeper look into quantum algorithms



## Postulate 1. Definition of qubit

In quantum computing, information is stored, manipulated and measured in the form of qubits.

A qubit may be mathematically represented as a unit vector in a two-dimensional Hilbert space  $|\psi\rangle \in \mathcal{H}^2$ .

A qubit  $|\psi\rangle$  may be written in general form as

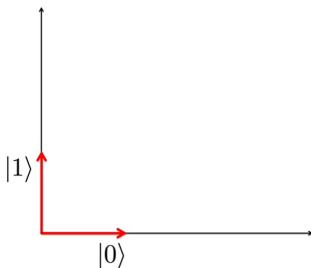
$$|\psi\rangle = \alpha|p\rangle + \beta|q\rangle$$

where  $\alpha, \beta \in \mathbb{C}$ ,  $|\alpha|^2 + |\beta|^2 = 1$  and  $\{|p\rangle, |q\rangle\}$  is an arbitrary basis spanning  $\mathcal{H}^2$ .

# Postulate 1. Definition of qubit

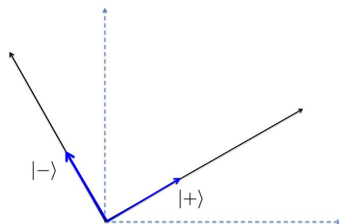
A most important consequence of the vectorial nature of a qubit is the possibility of writing it as a linear combination of elements of any basis.

For example, we may choose  $|p\rangle = |0\rangle$  and  $|q\rangle = |1\rangle$  to write a qubit. The basis  $\{|0\rangle, |1\rangle\}$  is known as **the computational basis**.



# Postulate 1. Definition of qubit

We may also choose the **the diagonal basis**  $\{|+\rangle, |-\rangle\}$  to write a qubit.



## Postulate 1. Definition of qubit

Choosing a concrete vector basis and values for corresponding complex coefficients is known as **preparing a qubit**. For instance,

$$|\psi\rangle = \frac{\sqrt{3}}{2}|0\rangle + \frac{i}{2}|1\rangle$$

In computer science parlance, preparing a qubit (or a set of qubits) is equivalent to variable initialization in a computer program.



## Postulate 2. Evolution by Unitary Operator

The evolution of a closed quantum system with state vector  $|\Psi\rangle$  is described by a Unitary operator<sup>1</sup>. The state of a system at time  $t_2$  according to its state at time  $t_1$  is given by

$$|\Psi(t_2)\rangle = \hat{U}|\Psi(t_1)\rangle.$$

Postulate 2 only describes the mathematical properties that an evolution operator must have. The specific evolution operator required to describe the behaviour of a particular quantum system depends on the system itself.

<sup>1</sup>Let  $\mathcal{H}$  be a Hilbert space and  $\hat{U} : \mathcal{H} \rightarrow \mathcal{H}$  a linear operator.  $\hat{U}$  is a **unitary operator** if  $\hat{U}\hat{U}^\dagger = \hat{U}^\dagger\hat{U} = \hat{I}$ , where  $\hat{U}^\dagger$  is the conjugate transpose of  $\hat{U}$  and  $\hat{I}$  is the identity operator.

## Postulate 2. Evolution by Schrödinger equation

Quantum evolution can also be written in terms of differential equations.

The time evolution of a closed quantum system can be described by the Schrödinger equation:

$$i\hbar \frac{\partial |\psi(t)\rangle}{\partial t} = \hat{\mathbf{H}} |\psi(t)\rangle$$

where  $\hbar = \frac{h}{\pi}$ ,  $h$  is Plank Constant, and  $\hat{\mathbf{H}}$  is a Hermitian operator known as the *Hamiltonian* of the sytem.

# Examples of Unitary Operators (1/2)

## Pauli operators

$$\hat{\sigma}_x = |0\rangle\langle 1| + |1\rangle\langle 0|; \hat{\sigma}_y = i|0\rangle\langle 1| - i|1\rangle\langle 0|; \hat{\sigma}_z = |0\rangle\langle 0| - |1\rangle\langle 1|$$

$$\text{If } |0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \langle 0| = (1, 0), \text{ and } \langle 1| = (0, 1)$$

Then we produce the following matrix representations of Pauli operators:

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad ; \quad \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \quad ; \quad \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

## Examples of Unitary Operators (2/2)

### The **Hadamard operator**

$$\hat{H} = \frac{1}{\sqrt{2}}(|0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| - |1\rangle\langle 1|)$$

Again, if  $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ ,  $|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ ,  $\langle 0| = (1, 0)$ , and  $\langle 1| = (0, 1)$

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

# (Already known) Example of Evolution by Unitary Operator

Let  $\hat{\sigma}_y = i|0\rangle\langle 1| - i|1\rangle\langle 0|$  and  $|\psi\rangle = \frac{\sqrt{3}}{2}|0\rangle + \frac{i}{2}|1\rangle$ . Compute  $\hat{\sigma}_y|\psi\rangle$ .

$$\begin{aligned}
 \hat{\sigma}_y|\psi\rangle &= (i|0\rangle\langle 1| - i|1\rangle\langle 0|)(\frac{\sqrt{3}}{2}|0\rangle + \frac{i}{2}|1\rangle) \\
 &= \frac{\sqrt{3}i\langle 1|0\rangle}{2}|0\rangle + \frac{i^2\langle 1|1\rangle}{2}|0\rangle - \frac{\sqrt{3}i\langle 0|0\rangle}{2}|1\rangle - \frac{i^2\langle 0|1\rangle}{2}|1\rangle \\
 &= -\frac{1}{2}|0\rangle - \frac{\sqrt{3}i}{2}|1\rangle
 \end{aligned}$$

## Postulate 3. Quantum Measurement

In quantum mechanics, measurement is a non-trivial and highly counter-intuitive process because:

**a) Measurement outcomes are inherently probabilistic.** Regardless of the carefulness in the preparation of a measurement procedure, the possible outcomes of such measurement are distributed according to a certain probability distribution.

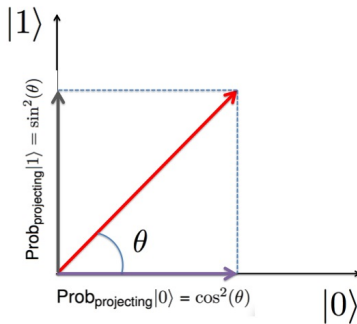
## Postulate 3. Quantum Measurement

### **b) Pre- and post-measurement quantum states are different.**

Once a measurement has been performed, a quantum system is unavoidably altered due to the interaction with the measurement apparatus. Consequently, for an arbitrary quantum system, pre-measurement and post-measurement quantum states are different in general.

## Postulate 3. Quantum Measurement

**c) Measuring  $\Leftrightarrow$  Projecting.** Measuring a quantum system is equivalent to projecting (as in analytic geometry) a vector onto a vector basis.





## Postulate 3. Quantum Measurement

The **KEY** points to remember are:

- Measuring a quantum system is a **probabilistic process** that changes the state of the quantum system.
- Extracting information from a quantum system (e.g. a photon) makes irreversible changes in the actual information contained in that quantum system.
- Measuring a quantum system is equivalent to projecting a vector onto a vector basis.

## Postulate 3. Quantum Measurement

**Example.** Let  $|\psi\rangle$  be a qubit representing a two-dimensional quantum-mechanical system (e.g., photon polarization). Let us write  $|\psi\rangle$  as

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

where  $||\alpha||^2 + ||\beta||^2 = 1$  and  $\{|0\rangle, |1\rangle\}$  is the computational basis spanning  $\mathcal{H}^2$ .

## Postulate 3. Quantum Measurement

According to Postulate 3, the probabilities of obtaining outcome  $a_0$  or outcome  $a_1$  are given by

$$p(a_0) = ||\alpha||^2 \text{ and } p(a_1) = ||\beta||^2$$

Corresponding post-measurement quantum states are as follows:

If outcome =  $a_0$  then  $|\psi\rangle_{pm} = |0\rangle$

If outcome =  $a_1$  then  $|\psi\rangle_{pm} = |1\rangle$ .

## Postulate 4. Composite quantum systems

We now focus on the mathematical description of a composite quantum system, i.e. **a system made up of several different physical systems**.

The state space of a composite quantum system is the tensor product of the component system state spaces.

If we have  $n$  quantum systems expressed as *state vectors*, labeled  $|\psi\rangle_1, |\psi\rangle_2, \dots, |\psi\rangle_n$  then the joint state of the total system is given by  $|\psi\rangle_T = |\psi\rangle_1 \otimes |\psi\rangle_2 \otimes \dots \otimes |\psi\rangle_n$ .

**Please note this crucial property:**  $\bigotimes_{i=1}^n \mathcal{H}_i^2 = \mathcal{H}^{2^n}$ .

## Postulate 4. Composite quantum systems

**Example 1.** Let  $\hat{H}^{\otimes 2}$  be the tensor product of the Hadamard operator with itself and let  $|\psi\rangle = |0\rangle \otimes |0\rangle = |00\rangle$ . Then

$$\begin{aligned}
 \hat{H}^{\otimes 2} |\psi\rangle &= \frac{1}{2}(|00\rangle\langle 00| + |01\rangle\langle 00| + |10\rangle\langle 00| + |11\rangle\langle 00| + |00\rangle\langle 01| \\
 &\quad - |01\rangle\langle 01| + |10\rangle\langle 01| - |11\rangle\langle 01| + |00\rangle\langle 10| + |01\rangle\langle 10| \\
 &\quad - |10\rangle\langle 10| - |11\rangle\langle 10| + |00\rangle\langle 11| - |01\rangle\langle 11| - |10\rangle\langle 11| \\
 &\quad + |11\rangle\langle 11|)|00\rangle \\
 &= \frac{1}{2}(\langle 00|00\rangle|00\rangle + \langle 00|00\rangle|01\rangle + \langle 00|00\rangle|10\rangle + \langle 00|00\rangle|11\rangle) \\
 &= \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)
 \end{aligned}$$

That is, the notion of inner product on orthonormal bases easily extends to  $\mathcal{H}^{2^n}$ .

## Postulate 4. Composite quantum systems

### Example 2. Introducing $\hat{C}_{\text{not}}$ .

Let us introduce  $\hat{C}_{\text{not}}$ , a two-qubit gate that behaves as follows:

$$\hat{C}_{\text{not}} = |00\rangle\langle 00| + |01\rangle\langle 01| + |10\rangle\langle 11| + |11\rangle\langle 10|$$

Then

$$\hat{C}_{\text{not}}|00\rangle = |00\rangle$$

$$\hat{C}_{\text{not}}|01\rangle = |01\rangle$$

$$\hat{C}_{\text{not}}|10\rangle = |11\rangle$$

$$\hat{C}_{\text{not}}|11\rangle = |10\rangle$$

The operator  $\hat{C}_{\text{not}}$  acts as a **Controlled Not**:  $\hat{C}_{\text{not}}$  flips the second qubit iff the first qubit is  $|1\rangle$ , otherwise  $\hat{C}_{\text{not}}$  behaves like a buffer.

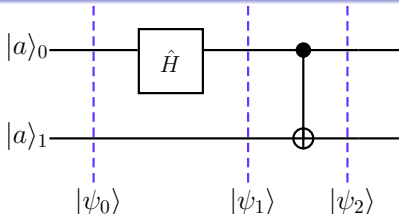
# Table of Contents

- 1 Thanks
- 2 Our goal
- 3 Who we are - QIP Group at Tecnológico de Monterrey
- 4 Quantum Computation -Definition and RDI ecosystem
- 5 Motivation and initial concepts - Quantum algorithms
- 6 A concise intro to Theoretical Computer Science
- 7 Mathematical preliminaries for Quantum Computation
- 8 Postulates of Quantum Mechanics
- 9 Our First Quantum Circuit!**
- 10 A deeper look into quantum algorithms





# Bell State Circuit (1/4)



Let

$$|a\rangle_0 = |0\rangle \text{ and } |a\rangle_1 = |0\rangle$$

Also, remember that

$$\hat{H} = \frac{1}{\sqrt{2}}(|0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| - |1\rangle\langle 1|)$$

$$\hat{C}_{\text{not}} = |00\rangle\langle 00| + |01\rangle\langle 01| + |10\rangle\langle 11| + |11\rangle\langle 10|$$

## Bell State Circuit (2/4)

So,

$$|\psi\rangle_0 = |0\rangle \otimes |0\rangle = |00\rangle$$

$$|\psi\rangle_1 = (\hat{H} \otimes \hat{I})(|0\rangle \otimes |0\rangle) = \hat{H}|0\rangle \otimes \hat{I}|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes |0\rangle$$

$$= \frac{1}{\sqrt{2}}|0\rangle|0\rangle + \frac{1}{\sqrt{2}}|1\rangle|0\rangle$$

$$|\psi\rangle_2 = \hat{C}_{\text{not}}(\frac{1}{\sqrt{2}}|0\rangle|0\rangle + \frac{1}{\sqrt{2}}|1\rangle|0\rangle)$$

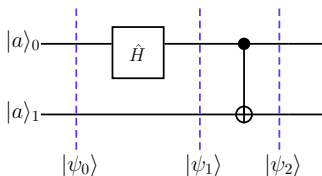
$$= \frac{1}{\sqrt{2}}|0\rangle|0\rangle + \frac{1}{\sqrt{2}}|1\rangle|1\rangle = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$$

So,

$$|\psi\rangle_2 = \frac{|00\rangle + |11\rangle}{\sqrt{2}}$$

# Bell State Circuit (3/4)

## Exercise



Compute  $|\psi\rangle_2$  for

$$|a\rangle_0 = |0\rangle \text{ and } |a\rangle_1 = |1\rangle$$

$$|a\rangle_0 = |1\rangle \text{ and } |a\rangle_1 = |0\rangle$$

$$|a\rangle_0 = |1\rangle \text{ and } |a\rangle_1 = |1\rangle$$

# Bell State Circuit (4/4)

## Answers

$$\begin{aligned}\hat{C}_{\text{not}}((\hat{H} \otimes \hat{I})|00\rangle) &= \frac{|00\rangle + |11\rangle}{\sqrt{2}} \\ \hat{C}_{\text{not}}((\hat{H} \otimes \hat{I})|01\rangle) &= \frac{|01\rangle + |10\rangle}{\sqrt{2}} \\ \hat{C}_{\text{not}}((\hat{H} \otimes \hat{I})|10\rangle) &= \frac{|00\rangle - |11\rangle}{\sqrt{2}} \\ \hat{C}_{\text{not}}((\hat{H} \otimes \hat{I})|11\rangle) &= \frac{|01\rangle - |10\rangle}{\sqrt{2}}\end{aligned}$$

These states are known as the **Bell states**

$$\begin{aligned}|\Phi^+\rangle &= \frac{|00\rangle + |11\rangle}{\sqrt{2}} \\ |\Phi^-\rangle &= \frac{|00\rangle - |11\rangle}{\sqrt{2}} \\ |\Psi^+\rangle &= \frac{|01\rangle + |10\rangle}{\sqrt{2}} \\ |\Psi^-\rangle &= \frac{|01\rangle - |10\rangle}{\sqrt{2}}\end{aligned}$$

## Quantum Entanglement (1/2)

Bell states are examples of entangled states. Bell states are key features of a quantum information transmission protocol known as quantum teleportation.

Quantum entanglement is a unique type of correlation shared between components of a quantum system.


Quantum entanglement and the principle of superposition are two of the main features behind the power of quantum computation and quantum information theory.

## Quantum Entanglement (2/2)

Entangled quantum systems are sometimes best used collectively, that is, sometimes an optimal use of entangled quantum systems for information storage and retrieval includes manipulating and measuring those systems as a whole, rather than on an individual basis.

- 1 Thanks
- 2 Our goal
- 3 Who we are - QIP Group at Tecnológico de Monterrey
- 4 Quantum Computation -Definition and RDI ecosystem
- 5 Motivation and initial concepts - Quantum algorithms
- 6 A concise intro to Theoretical Computer Science
- 7 Mathematical preliminaries for Quantum Computation
- 8 Postulates of Quantum Mechanics
- 9 Our First Quantum Circuit!
- 10 A deeper look into quantum algorithms**

# General structure of a quantum algorithm

	Classical algorithm	Quantum Algorithm
Information Unit	Bit $x \in \{0, 1\}$	Qubit $ \psi\rangle \in \mathbb{C}^2$
Algorithm design	$\{\text{AND, OR, NOT}\}$ Automata (e.g. Turing Machines) C, Python	$ \Psi_{t_2}\rangle = \hat{U} \Psi_{t_1}\rangle$ $i\hbar \frac{d \psi\rangle}{dt} = \hat{H} \psi\rangle$ 
Outcome	Memory readout	Quantum measurement

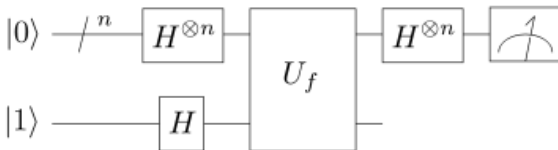


- State of the art on quantum programming languages:
  - Quantum gate-based graphical interface — IBM.
  - Python, C, C++ plus APIs — D-Wave
  - Full quantum programming languages: yet to come 😊
- Software for digital simulation of quantum algorithms
  - IBM. Q interface and QSKIT (Python)
  - Microsoft. Liquid and Q#
  - D-Wave. SAPI.

There are several methods to write a quantum algorithm.

# Quantum gates

A most popular method to write a quantum algorithm is to use quantum gates/circuits, which are mathematical representations of physical transformations on quantum systems.

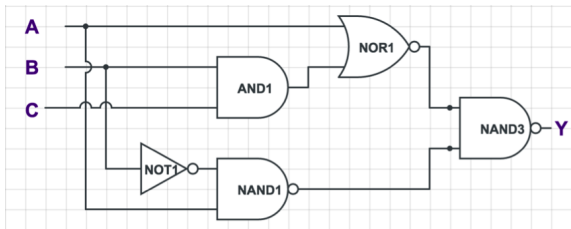


Deutsch-Jozsa quantum algorithm. Source: Wikipedia ©

This programming method is based on  $|\Psi_{t_2}\rangle = \hat{U}|\Psi_{t_1}\rangle$ .

# Quantum gates

The rationale behind the quantum gate method is very similar to the one followed in the construction of digital circuits:



A digital circuit employing NAND, NOR, NOT and AND gates. Source: Wikipedia ©

- The mathematical description of digital circuitry is determined and constrained by the definitions and results of mathematical logic.
- As for quantum circuits, the mathematical description of a quantum gate is determined by the physics of quantum evolution (that is, the behaviour of quantum systems in time).

# Schrödinger equation

We may also use differential equations to write quantum algorithms:

$$i\hbar \frac{d|\psi\rangle}{dt} = \hat{\mathbf{H}}|\psi\rangle$$

This is the Schrödinger equation and it is basic equation in the models of quantum annealing and adiabatic quantum computation.

## Example - Grover's Algorithm

A quantum algorithm developed by Lov Grover, its purpose is to perform a search of an item from an **unsorted and unstructured** list of  $N$  records.

- A classical computer brute-force *sequential* algorithm would require  $O(N)$  steps to perform the search.
- In contrast, Grover's algorithm requires only  $O(\sqrt{N})$  steps.
- Key concepts: quantum superposition, quantum parallelism and quantum oracle-based computation.

## Example - Shor's Algorithm

A quantum algorithm developed by Peter Shor, its purpose is to efficiently compute the prime factors of an  $N$ -digit integer ( $N = 2^n$ ).

- A classical computer brute-force algorithm would require  $O(2^{n/2})$  steps to perform the search.
- Quantum solution: Shor's algorithm,  $O(n^3)$  steps.
- Key concept: Shor's algorithm provides exponential acceleration wrt **known** classical algorithms.