# Introduction to Quantum Computation - PHY 5650

Eduardo Mucciolo (MAP 416, 3-1882, mucciolo@physics.ucf.edu)

Classes: *Mon, Wed, Fri* (10:30-11:20am) - MAP 204

Textbook: *Quantum Computation and Quantum Information*, by Michael A. Nielsen and Isaac L. Chuang (Cambridge University Press, 2000)

Grading: 50% problem sets + 50% final paper

Course web page: http://www.physics.ucf.edu/~mucciolo/phy5650/

# CALENDAR –– SPRING 2005

### January

| M | W | F |
|---|---|---|
| 03 | 05 | 07 |
| 10 (1) | 12 (2) | 14 (3) |
| 17 | 19 (4) | 21 (5) |
| 24 | 26 (6) | 28 (7) |
| 31 (8) | | |

### February

| M | W | F |
|---|---|---|
| | 02 (9) | 04 (10) |
| 07 (11) | 09 (12) | 11 (13) |
| 14 | 16 (14) | 21 (15) |
| 21 (16) | 23 (17) | 25 (18) |
| 28 (19) | | |

### March

| M | W | F |
|---|---|---|
| | 02 (20) | 04 (21) |
| 07 (22) | 09 (23) | 11 (24) |
| 14 | 16 | 21 |
| 21 | 23 | 25 |
| 28 (25) | 30 (26) | |

### April

| M | W | F |
|---|---|---|
| | | 01 (27) |
| 04 (28) | 06 (29) | 08 (30) |
| 11 (31) | 13 (32) | 15 (33) |
| 18 (34) | 20 (35) | 22 (36) |
| 25 (37) | 27 | 29 |

Caption:

**21 = Spring break** **14 = No class**

**17 = Holiday** 10 (1) **= Regular class day (#1)**

**= Problem set due**

# Course Content

1) *Motivation and Overview*: history, quantum bits (qubits), quantum circuits, quantum computation, quantum algorithms.

2) *Classical Computation*: Turing machines, computational complexity, complexity classes.

3) *The Basics of Quantum Mechanics*: linear algebra, postulates of Quantum Mechanics, superposition, interference, entanglement, time evolution, phase coherence.

4) *Quantum Circuits*: qubit operations and quantum gates, universal quantum gates.

5) *Quantum Computation*: quantum computational complexity, quantum algorithms, Shor's factorization algorithm, search algorithms.

6) *Physical Implementations*: optical and atomic, nuclear (NMR), solid state; scalability; the decoherence problem.

*Left out: error correction, quantum information*

# 1.1 Motivation

What is computation?

The implementation of an algorithm to perform a certain task (usually to solve a problem, calculate a function, etc).

The modern definition of a "computer" was born in 1936 with the advent of a model known as the Universal Turing Machine: a machine that implements an algorithm by following a set of instructions recorded in some sort of hardware (tape?), one at a time. It is programable, but needs to satisfy certain conditions.

Anything that is computable by some piece of hardware is also computable by a Universal Turing Machine (*the Church-Turing thesis*).

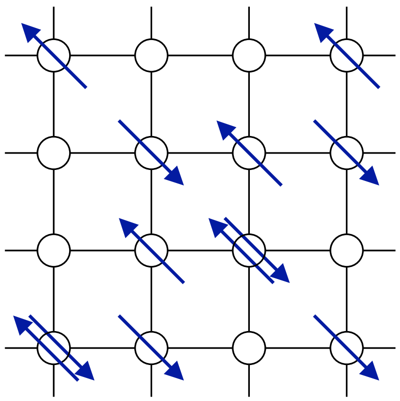
Notice: this is a very *physical* notion!

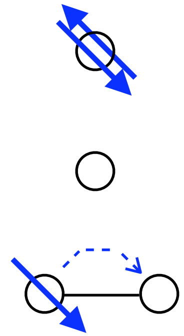
OBS: Not everything is algorithmically computable.

The catch:

Not all problems can be efficiently solved by a Universal Turing Machine.

Ex: The 2-dimesional Hubbard model (high-Tc superconductivity?)





$E = U$

$E = 0$

$E = -t$

*Find all eigenvalues and eigenfunctions for a 1000 x 1000 lattice.*

A computable problem... but currently impractical (the best we can do is 4 x 4).

Ex: Factorize 9046635314729965676224664254051240722094475168481918599
8005137843414469489753806381722276074208698889835256178
881805832120802638900447213803801057153472546

(Even if you have access to many fast computers, it may still take several days to do it - it took 8 months and 600 people (1,600 computers) to factorize a slightly smaller integer in 1994.)

In the 1980’s, people began to search for new paradigms for computation, beyond the Turing Machine.

Feynman: Simulating a quantum system with a conventional computer requires an exponentially large amount of resources! Use another *quantum system* to do the job instead.

Deutsch: Use the laws of physics - quantum mechanics - to define a new, more powerful and efficient Turing machine.

## Why Quantum Mechanics?

1) Computation deals with information.

2) Information is physical.

3) Quantum Mechanics is the mathematical framework that underlies all physical theories.

**Nature is quantum mechanical!**

The rules provide by Quantum Mechanics describe physical phenomena at all length scales, from the behavior of elementary particles to the structure of the universe.

Conventional computers may be able to simulate classical problems (turbulence, diffusion, etc.) efficiently, but it will take a computer based on quantum mechanics to simulate a quantum system with tangible resources (time, memory, etc).

Quantum Computation began to take form in the mid 1980's, but the "quantum leap" only happened in 1994, with the demonstration of a "killer application" by Shor: A quantum algorithm that factorizes any integer efficiently.

# 1.2 Qubits

classical bit: $\boxed{c} = (0) \text{ or } (1)$

binary representation:

$$\begin{aligned} 13 &= 2 \times 6 + 1 \\ &= 2 \times 2 \times 3 + 1 \\ &= 2 \times 2 \times 2 + 2 \times 2 + 1 \\ &= 2^3 + 2^2 + 2^0 = \begin{array}{|c|c|c|c|} \hline 1 & 1 & 0 & 1 \\ \hline \end{array} \\ & \qquad\qquad\qquad\qquad\quad\; \begin{array}{cccc} 3 & 2 & 1 & 0 \end{array} \end{aligned}$$

$$\begin{array}{r} + \begin{array}{|c|c|} \hline 0 & 1 \\ \hline 0 & 1 \\ \hline \end{array} \\ \hline \begin{array}{|c|c|} \hline 1 & 0 \\ \hline \end{array} \end{array}$$

quantum bit: $\boxed{q} = \alpha \times (0) + \beta \times (1)$

*superposition*

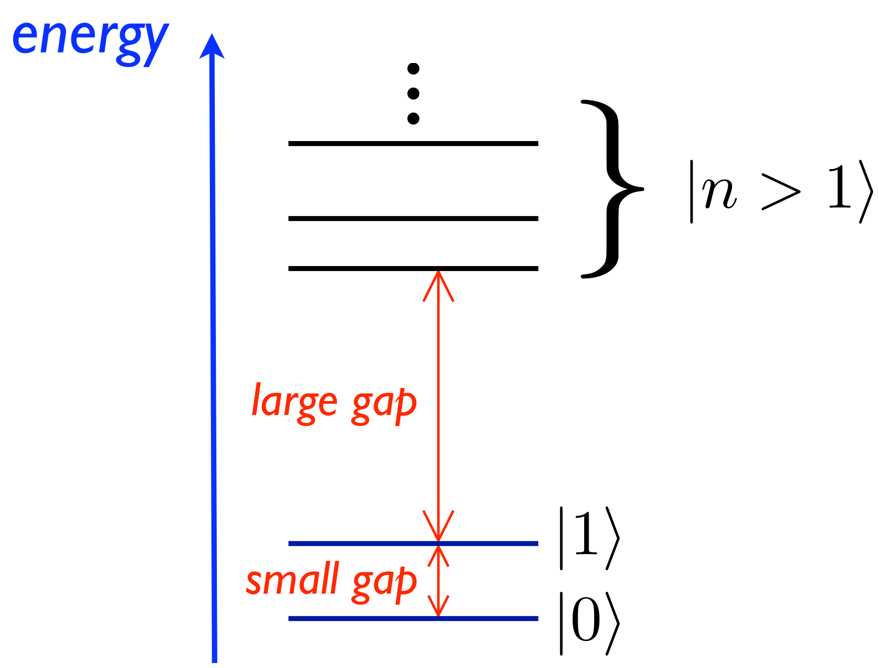
$$\alpha, \beta = \text{complex numbers} \; (|\alpha|^2 + |\beta|^2 = 1)$$

*Dirac notation:* $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$

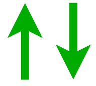
*"state vector"*

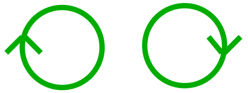
*Any decent qubit lives in a 2-dimensional Hilbert space!*

Physical definition: Two isolated, distinct configurations ("states") of a quantum system.




*Candidates*: atomic states, electron and nuclear spins, photon polarizations, ...



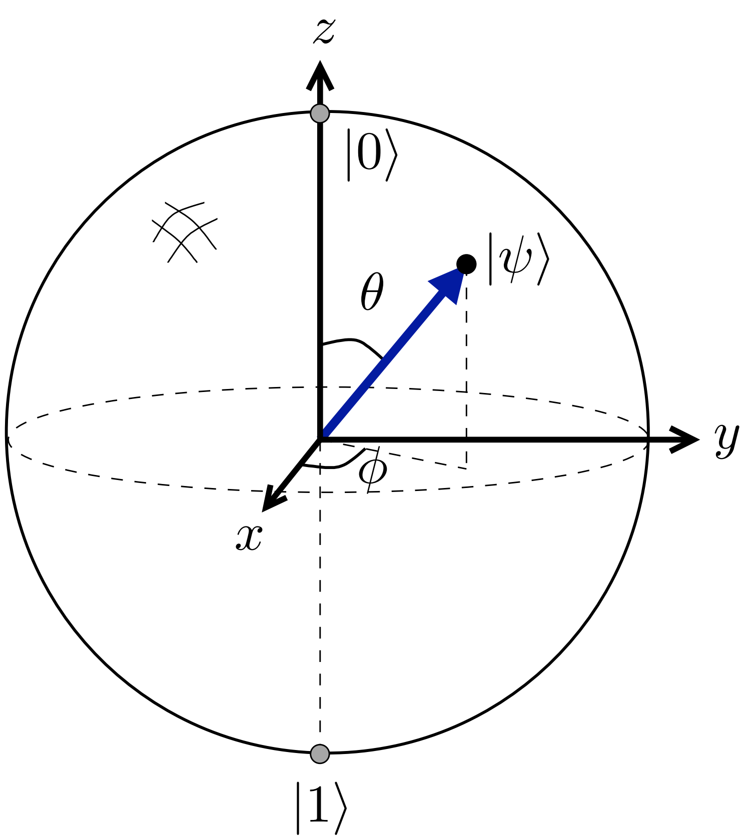


0B

# The Bloch sphere representation

$$|\alpha|^2 + |\beta|^2 = 1 \Rightarrow \begin{cases} \alpha = e^{i\gamma}\cos(\theta/2) \\ \beta = e^{i\lambda}\sin(\theta/2) \end{cases}$$

*complex amplitudes*




$$|\psi\rangle = \underbrace{e^{i\gamma}}_{\text{irrelevant}}\Big[\cos(\theta/2)|0\rangle + e^{i\phi}\sin(\theta/2)|1\rangle\Big]$$

$$\phi = \lambda - \gamma$$

$$\boxed{\begin{aligned} |0\rangle &\rightarrow \theta = 0 \ (\text{“}z\text{”}) \\ |1\rangle &\rightarrow \theta = \pi \ (\text{“}-z\text{”}) \end{aligned}}$$

$$\text{“}x\text{”} \rightarrow \theta = \pi/2,\ \phi = 0 \longrightarrow \frac{1}{\sqrt{2}}\Big(|0\rangle + |1\rangle\Big)$$

$$\text{“}y\text{”} \rightarrow \theta = \pi/2,\ \phi = \pi/2 \longrightarrow \frac{1}{\sqrt{2}}\Big(|0\rangle + i\,|1\rangle\Big)$$

0C

Question: *How much information can you store in a qubit?*

<u>classical bit</u>: discrete (0 or 1)

<u>quantum bit</u>: continuous ($\phi$ and $\theta$) *infinite?*

Answer: *It depends!*

Quantum Mechanics:

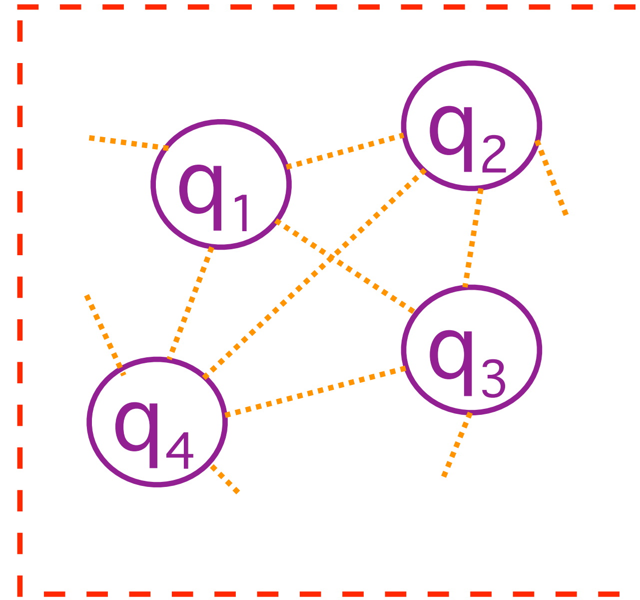
measuring $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$

$\rightarrow |0\rangle$ *probability* $|\alpha|^2$

$\rightarrow |1\rangle$ *probability* $|\beta|^2$

Quantum information is “hidden”

# Multiqubits:




- *direct product state:* $|\Psi\rangle = |\psi_1\rangle \otimes |\psi_2\rangle$

*Ex:* $|\Psi\rangle = |0\rangle_1 \otimes |0\rangle_2 \ (= |00\rangle)$

- *entangled state:* $|\Psi\rangle \neq |\psi_1\rangle \otimes |\psi_2\rangle$

*Ex:* $$|\Psi\rangle = \frac{1}{\sqrt{2}}\Big(|01\rangle + |10\rangle\Big)$$

EPR (Einstein-Podolsky-Rosen) pairs or Bell states: $$\begin{cases} \dfrac{1}{\sqrt{2}}\Big(|01\rangle \pm |10\rangle\Big) \\ \dfrac{1}{\sqrt{2}}\Big(|00\rangle \pm |11\rangle\Big) \end{cases}$$

*correlated!*

*General 2-qubit state:*

$$|\Psi\rangle = \alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle$$

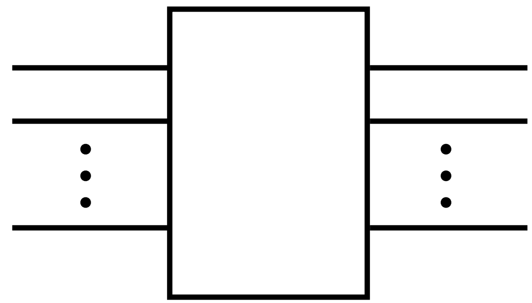
*Ex:* $$|00\rangle + |01\rangle + |10\rangle - |11\rangle = (|0\rangle + |1\rangle)_1(|0\rangle + |1\rangle)_2 - 2|11\rangle$$

0E

# 1.3 Quantum computation

## *Manipulating bits:* Logical gates

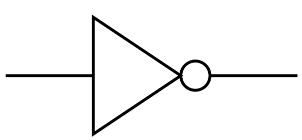
*input*



*output*

Ex: classical NOT gate

$0\ (1)$



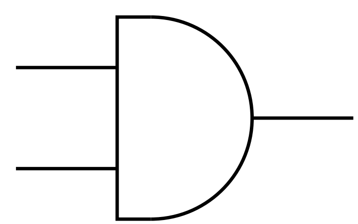
$1\ (0)$

$a$ $\qquad\qquad$ $\bar{a}$

Ex: classical AND gate

$a$

$b$



$c = a \text{ AND } b$

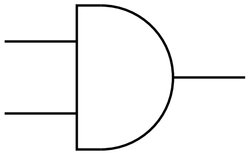
| $a$ | $b$ | $c$ |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |

truth table

0F

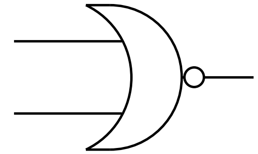
# 2-bit classical gates:

AND



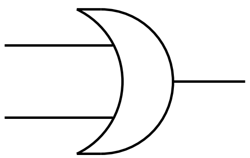
| $a$ | $b$ | $c$ |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |

NOR



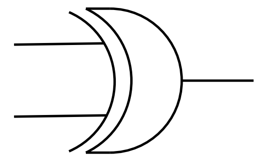
| $a$ | $b$ | $c$ |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 0 |

OR



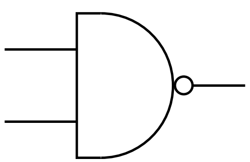
| $a$ | $b$ | $c$ |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

XOR



| $a$ | $b$ | $c$ |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 0 |

NAND

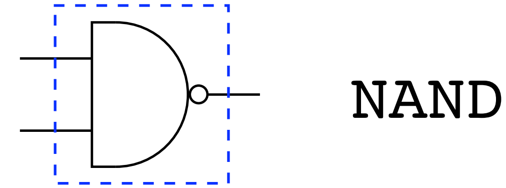


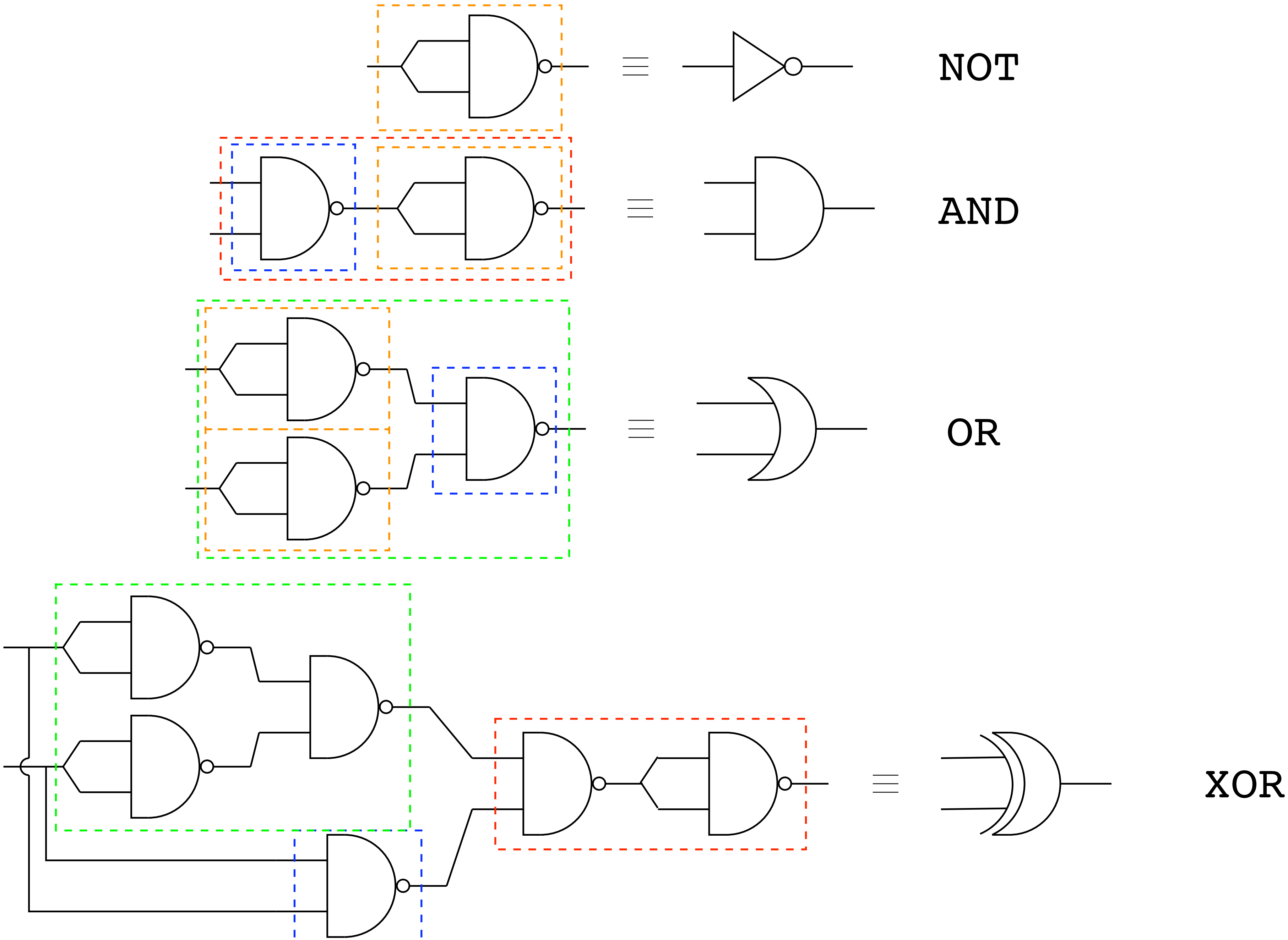
| $a$ | $b$ | $c$ |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 0 |

*Universal gate for classical computation*

*Obs*: They are useless for quantum computation: Irreversible!
However, it is possible to do classical computation reversibily.

# What does "universal gate" mean?

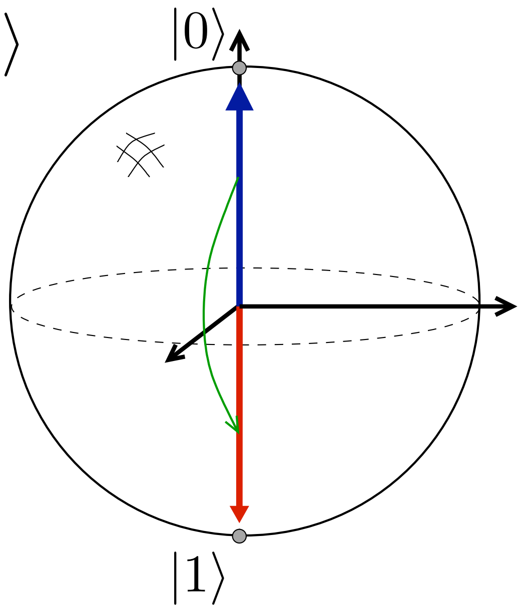


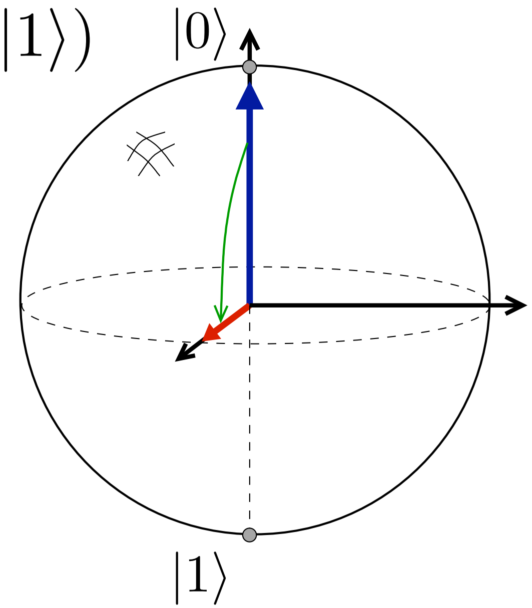
# 1-qubit quantum gates are rotations in the Bloch sphere:

$$X|0\rangle = |1\rangle$$

NOT operation




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




*(OBS: it is actually more complicated than it looks)*

# Quantum gates are unitary operations:

$U|\psi\rangle = |\psi'\rangle$ *with* $||\psi|| = ||\psi'||$ ⇨ $UU^\dagger = I$

*(check it!)*

# Quantum gates: matrix representation

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

NOT operation $$\underbrace{\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}}_{X \text{ gate}} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \beta \\ \alpha \end{pmatrix}$$

*Linear algebra!*

$$Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \qquad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \qquad H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

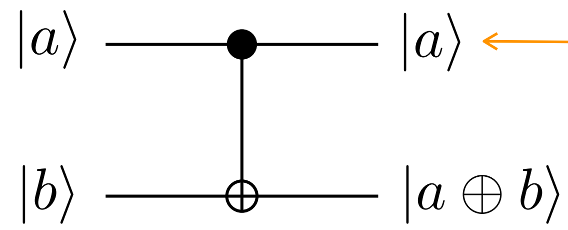
*no classical counterpart*

Hadamard ("square root of NOT")

# 2-qubit gates: Unitary operations $\Rightarrow$ Reversible

Ex: CNOT

equal number of input and output lines




*control*

*target*

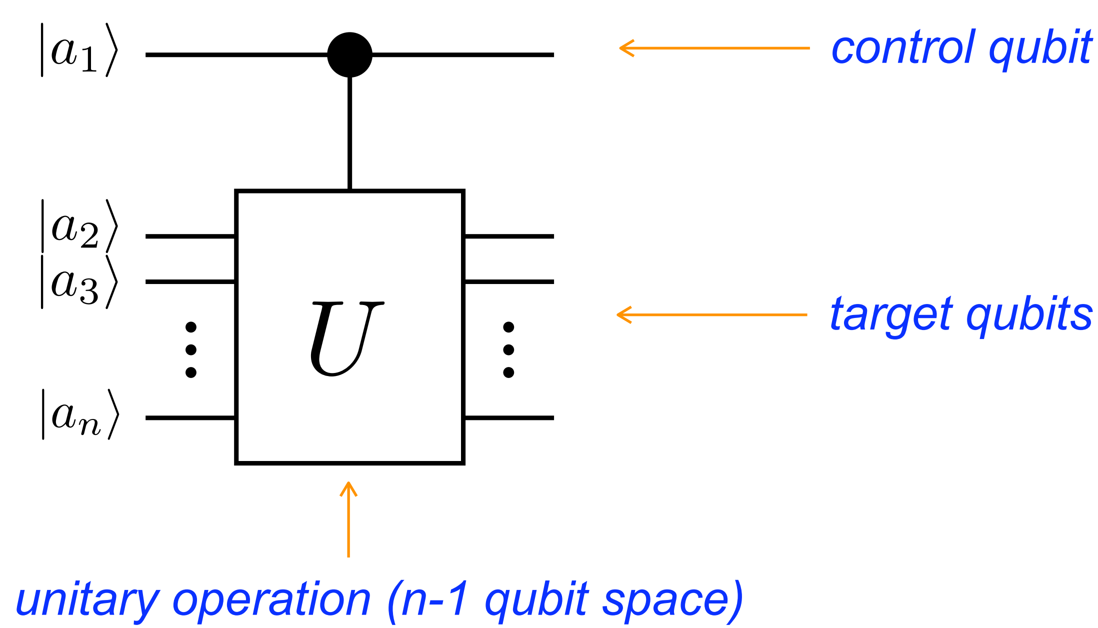
$$\boxed{x \oplus y = x + y \pmod 2 \qquad 0 \oplus 0 = 0 \qquad 0 \oplus 1 = 1 \qquad 1 \oplus 1 = 0}$$

$$|00\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \qquad |01\rangle = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \qquad |10\rangle = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \qquad |11\rangle = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

$$U_{\mathrm{CNOT}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

*Universal gate for quantum computation*

# Ex: Controlled *U* gate




***input:***

$$|a_1\rangle \otimes |a_2, a_3, \ldots, a_n\rangle$$

***output:***

$$|a_1\rangle \otimes |a_2, a_3, \ldots, a_n\rangle \qquad \textit{if} \quad a_1 = 0$$

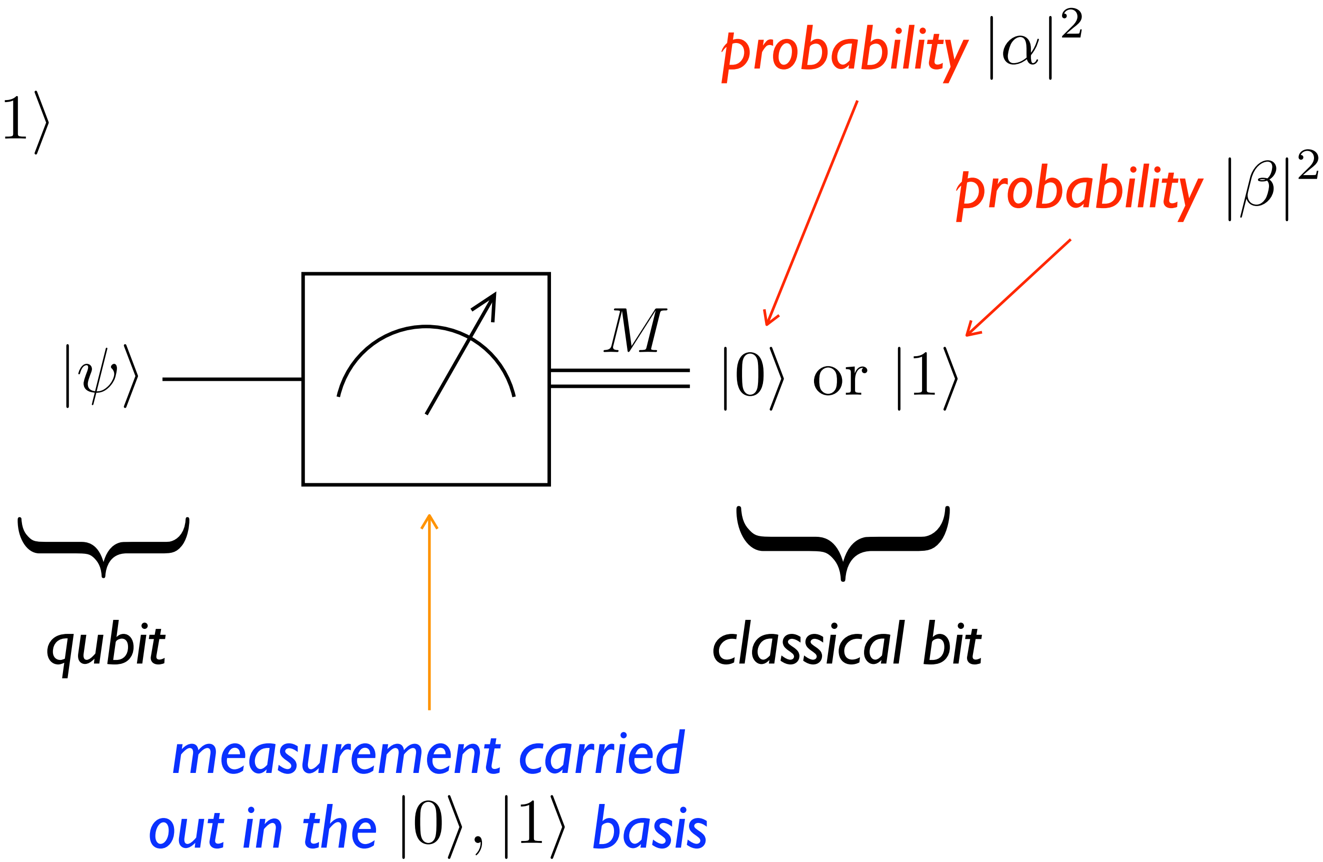
***or***

$$|a_1\rangle \otimes \big(U\, |a_2, a_3, \ldots, a_n\rangle\big) \qquad \textit{if} \quad a_1 = 1$$

*Obs:* CNOT is a special case ($n = 2,\ U = X$)

# Quantum measurement:

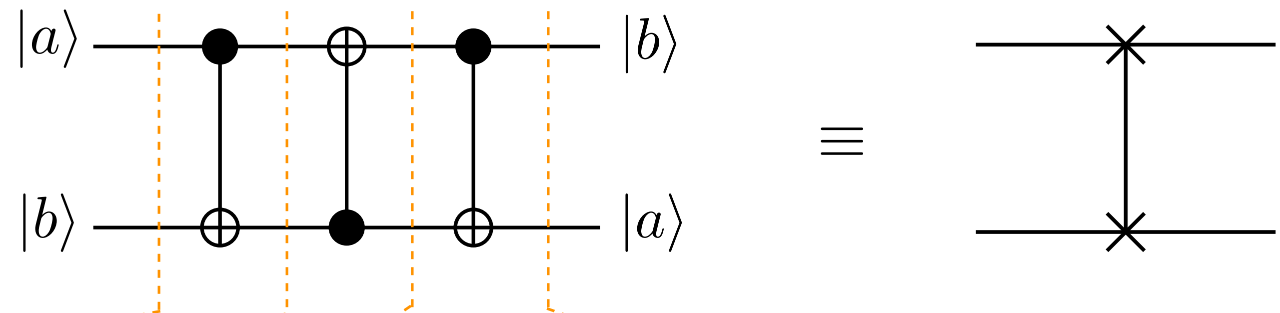
$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$




*Obs*: (1) Measurements can be carried out in *any* basis.
(2) The output is still a quantum state.
(3) Measurements destroys "hidden" information (unless it is repeated infinitely over identical states).

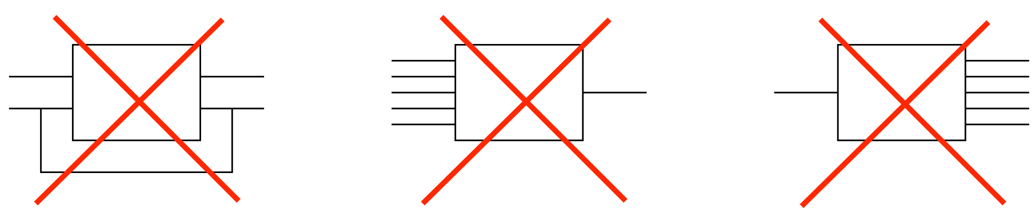
# Quantum circuits: sequences of unitary operations

Ex: *"swap" circuit*

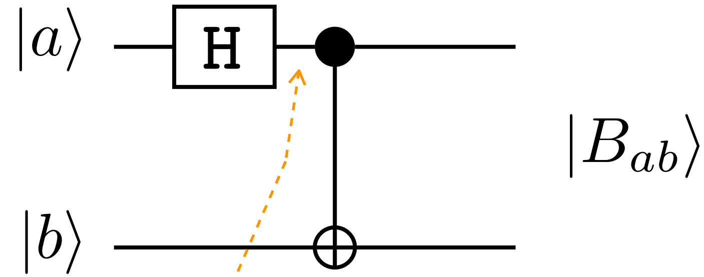



$$|a,b\rangle \to |a, a\oplus b\rangle \to |a\oplus a\oplus b, a\oplus b\rangle = |b, a\oplus b\rangle \to |b, a\oplus b\oplus b\rangle = |b,a\rangle$$

***Obs***: Feedback loops, fan-in and fan-out configurations are not allowed.



## Ex: *creating Bell states*




$$|a\rangle = \alpha|0\rangle + \beta|1\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \qquad H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

$$|c\rangle = H|a\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix} \alpha+\beta \\ \alpha-\beta \end{pmatrix} = \frac{1}{\sqrt{2}}\left[(\alpha+\beta)|0\rangle + (\alpha-\beta)|1\rangle\right]$$

$$U_{\mathrm{CNOT}}|c,b\rangle = |c, c\oplus b\rangle = \frac{1}{\sqrt{2}}\left[(\alpha+\beta)|0,b\rangle + (\alpha-\beta)|1, 1\oplus b\rangle\right]$$

$$\uparrow$$

$$1\oplus b = \bar{b}$$

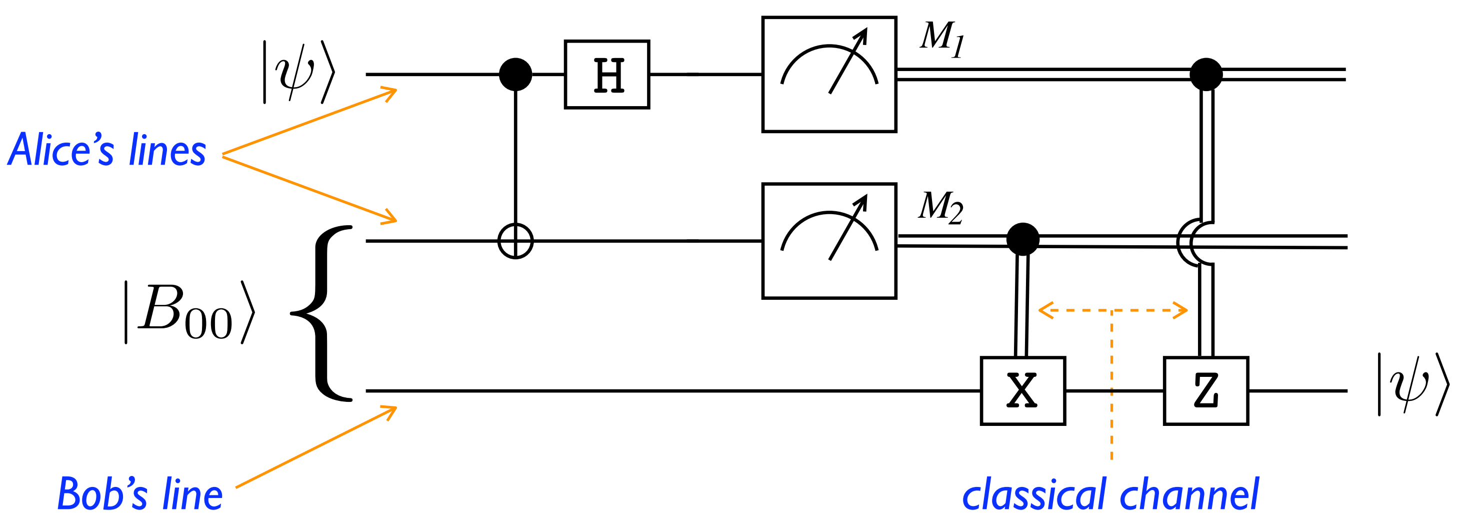
| | |
|---|---|
| $\lvert 00\rangle$ | $(\lvert 00\rangle + \lvert 11\rangle)/\sqrt{2} = B_{00}$ |
| $\lvert 01\rangle$ | $(\lvert 01\rangle + \lvert 10\rangle)/\sqrt{2} = B_{01}$ |
| $\lvert 10\rangle$ | $(\lvert 00\rangle - \lvert 11\rangle)/\sqrt{2} = B_{10}$ |
| $\lvert 11\rangle$ | $(\lvert 01\rangle - \lvert 10\rangle)/\sqrt{2} = B_{11}$ |

# Ex: *quantum teleportation*

The problem:

(1) Alice and Bob met. They shared an entangled state, $|B_{00}\rangle$.
(2) They were separated. Years went by.
(3) Alice now wants to send a certain qubit $|\psi\rangle$ to Bob, but:
 (i) she only has a single copy of it;
 (ii) they can only communicate through classical channels;
 (iii) any measurement she makes will destroy the "hidden" information.
(4) What should they do?

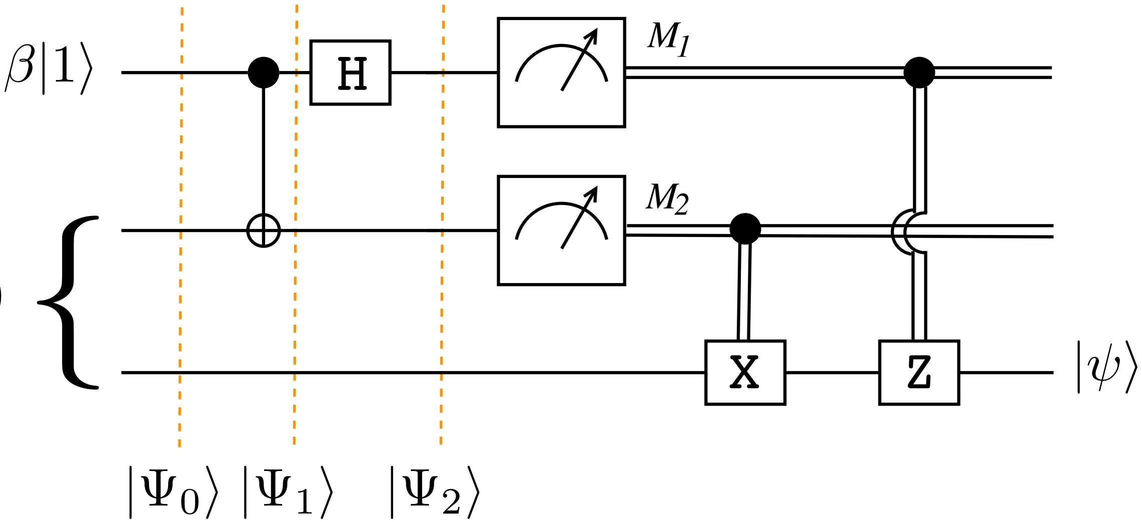
The solution:



# Step-by-step procedure:

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

$$|B_{00}\rangle = \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right)$$




$$|\Psi_0\rangle = |\psi\rangle|B_{00}\rangle = \frac{1}{\sqrt{2}}\left[\alpha(|0\rangle|00\rangle + |0\rangle|11\rangle) + \beta(|1\rangle|00\rangle + |1\rangle|11\rangle)\right]$$

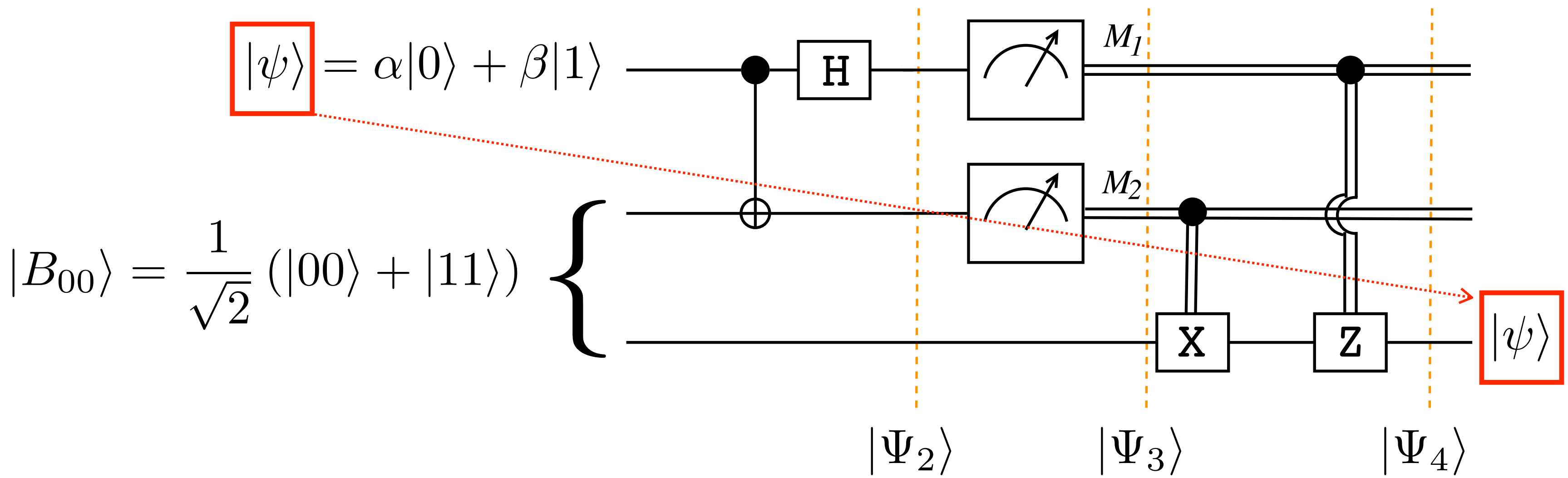
$$|\Psi_1\rangle = \frac{1}{\sqrt{2}}\left[\alpha(|0\rangle|00\rangle + |0\rangle|11\rangle) + \beta(|1\rangle|10\rangle + |1\rangle|01\rangle)\right]$$

$$|\Psi_2\rangle = \frac{1}{\sqrt{2}}\left[\frac{\alpha}{\sqrt{2}}(|0\rangle + |1\rangle)(|00\rangle + |11\rangle) + \frac{\beta}{\sqrt{2}}(|0\rangle - |1\rangle)(|10\rangle + |01\rangle)\right]$$

$$= \frac{1}{2}\left[|00\rangle(\alpha|0\rangle + \beta|1\rangle) + |01\rangle(\alpha|1\rangle + \beta|0\rangle) + |10\rangle(\alpha|0\rangle - \beta|1\rangle) + |11\rangle(\alpha|1\rangle - \beta|0\rangle)\right]$$

1A

# Step-by-step procedure:




$$|\Psi_2\rangle = \frac{1}{2}\left[|00\rangle(\alpha|0\rangle+\beta|1\rangle)+|01\rangle(\alpha|1\rangle+\beta|0\rangle)+|10\rangle(\alpha|0\rangle-\beta|1\rangle)+|11\rangle(\alpha|1\rangle-\beta|0\rangle)\right]$$

$$M_1 = 0,\ M_2 = 0 \rightarrow |\Psi_3\rangle = |00\rangle(\alpha|0\rangle + \beta|1\rangle) \longrightarrow |\Psi_4\rangle = |00\rangle|\psi\rangle$$

$$M_1 = 0,\ M_2 = 1 \rightarrow |\Psi_3\rangle = |01\rangle(\alpha|1\rangle + \beta|0\rangle) \xrightarrow{\text{X}} |\Psi_4\rangle = |01\rangle|\psi\rangle \qquad \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} \beta \\ \alpha \end{pmatrix} = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$$

$$M_1 = 1,\ M_2 = 0 \rightarrow |\Psi_3\rangle = |10\rangle(\alpha|0\rangle - \beta|1\rangle) \xrightarrow{\text{Z}} |\Psi_4\rangle = |10\rangle|\psi\rangle \qquad \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}\begin{pmatrix} \alpha \\ -\beta \end{pmatrix} = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$$

$$M_1 = 1,\ M_2 = 1 \rightarrow |\Psi_3\rangle = |11\rangle(\alpha|1\rangle - \beta|0\rangle) \xrightarrow{\text{X,Z}} |\Psi_4\rangle = |11\rangle|\psi\rangle \quad \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} -\beta \\ \alpha \end{pmatrix} = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$$

# 1.4 Quantum algorithms

Computation can be done by connecting circuit elements.

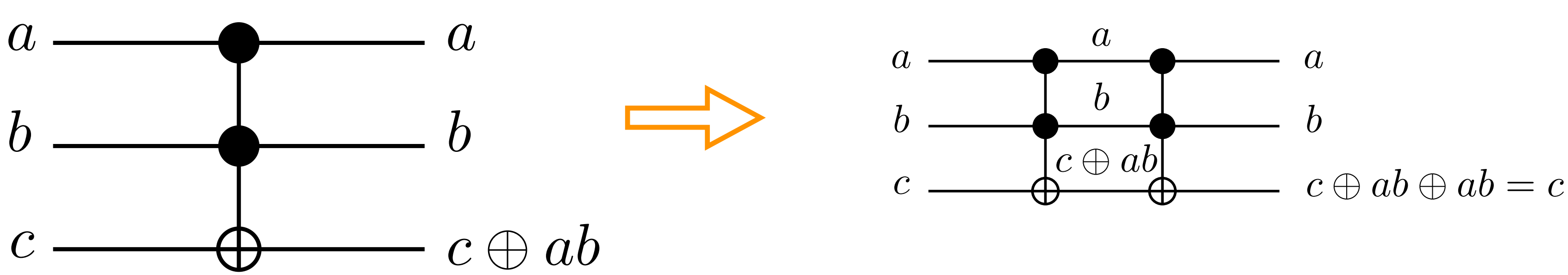
In order to do interesting things with a computer we need algorithms!

Q: Can a quantum computer perform all classical operations?

A: Yes!

Q: How?

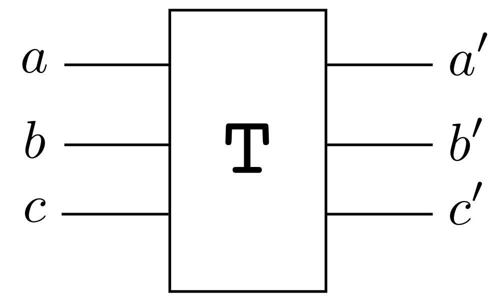
A: Reversible classical computation: the Toffoli gate.



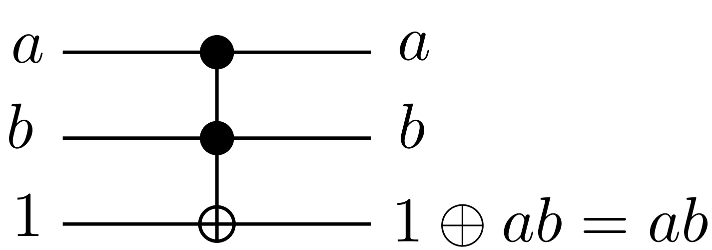
1C

*Truth table for the Toffoli gate*

| $a$ | $b$ | $c$ | $a'$ | $b'$ | $c'$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 0 |

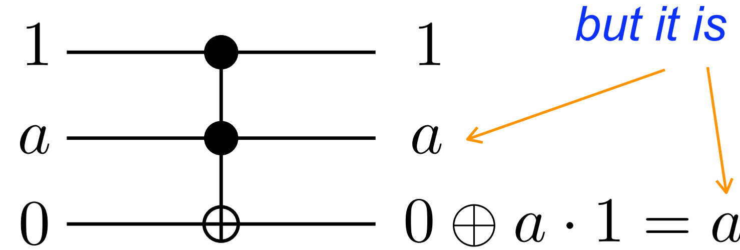



NAND implementation:




FANOUT implementation:

*looks like cloning, but it is not!*




*The Toffoli gate is a legitimate quantum gate as well.*

*(Find the unitary operator $U_{\text{Toffoli}}$.)*

# Quantum computers can perform all classical algorithms... so what?

*Q:* Can they outperform classical?

*A:* Yes, at least for certain algorithms (stay tuned!).

*Q:* Can they do something classical computers cannot?

*A:* Yes! <u>Quantum "parallelism":</u>

Suppose we want to compute $f(x)$ for $x = 0, 1, \ldots, 2^{n-1}$ ($n$ bits).

$$f : \{0, 1, \ldots, 2^n - 1\} \longrightarrow \{0, 1, \ldots, 2^{n-1}\}$$

$$x \longrightarrow f(x)$$

*number in binary basis*

Ex: 1-bit case. Calculate $f(0)$ and $f(1)$.

*Classical algorithm*

```
DO X=0,1
   F(X)=...function call...
END DO
```

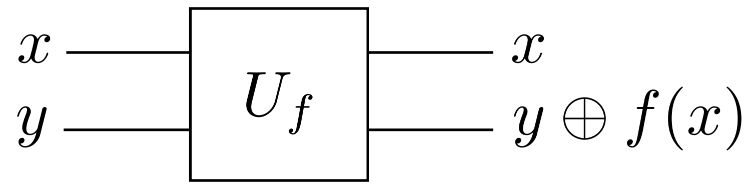
$\longleftarrow$ *2 steps*

1E

## *Quantum algorithm*

Suppose we have the unitary 2-qubit gate

$$U_f|x, y\rangle = |x, y \oplus f(x)\rangle$$




Take $|x\rangle = \dfrac{|0\rangle + |1\rangle}{\sqrt{2}}$ and $|y\rangle = |0\rangle$ . Then,

$$U_f\left(\frac{|00\rangle + |10\rangle}{\sqrt{2}}\right) = \frac{1}{\sqrt{2}}\big(|0, 0 \oplus f(0)\rangle + |1, 0 \oplus f(1)\rangle\big)$$
$$= \frac{1}{\sqrt{2}}\big(|0, f(0)\rangle + |1, f(1)\rangle\big)$$

*f(x) computed over the whole domain in 1 step*

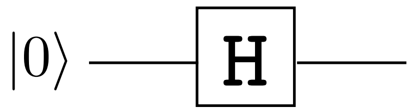
*Classical parallelism:* *N* circuits doing the same computation simultaneously.

*Quantum parallelism:* only *1* circuit doing *N* computations at once.

1F

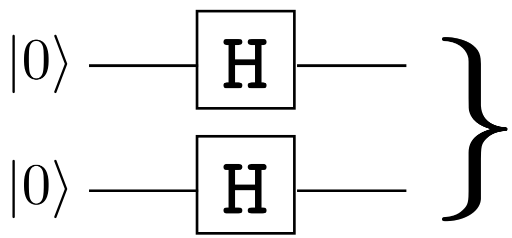
# General case:

(i) prepare the initial state *(use a Hadamard gate)*




$$\frac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big)$$

*1-qubit case*

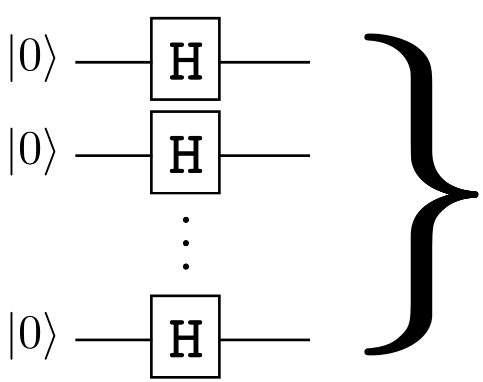



$$\frac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big) \otimes \frac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big)$$

*2-qubit case*

$$= \frac{1}{2}\big(|00\rangle + |10\rangle + |01\rangle + |11\rangle\big)$$

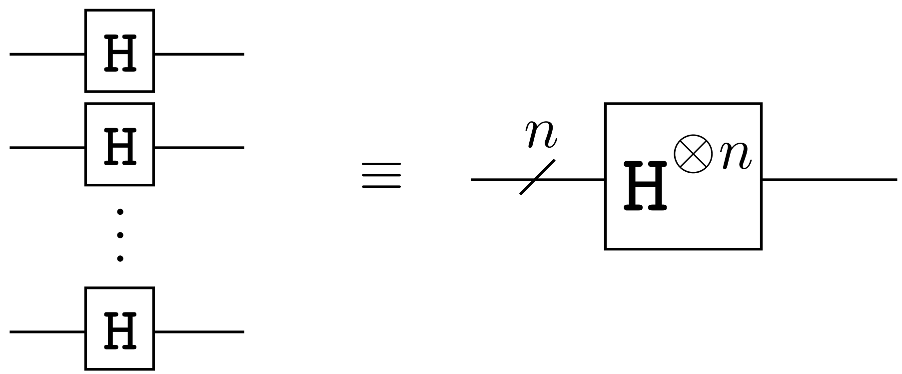
$$= \frac{1}{2}\sum_{x_1,x_2=0}^{1} |x_1 x_2\rangle$$




$$\frac{1}{2^{n/2}}\sum_{x_1,\ldots,x_n=0}^{1} |x_1 x_2 \ldots x_n\rangle = \frac{1}{2^{n/2}}\sum_{X=1}^{2^n-1} |X\rangle^n$$

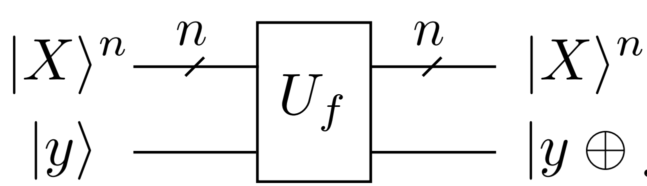
*n-qubit case*

# General case:

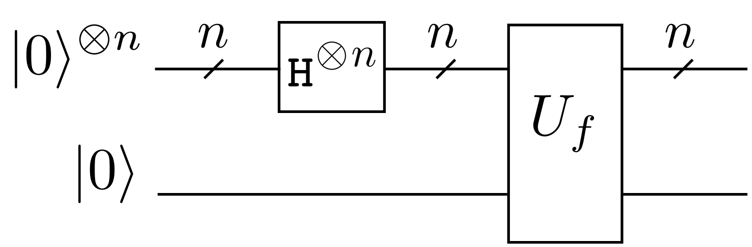



*n-dimensional Hadamard gate*

(ii) implementing the unitary transformation $f : \{0,1\}^n \longrightarrow \{0,1\}$




(iii) parallel computation




$$\frac{1}{2^{n/2}} \sum_{X=1}^{2^n-1} |X\rangle^n$$

$$\frac{1}{2^{n/2}} \sum_{X=1}^{2^n-1} |f(X)\rangle$$

*output*

$$\boxed{\frac{1}{2^{n/2}} \sum_{X=1}^{2^n-1} |X, f(X)\rangle^{n+1}}$$

# Caveat:

$$\boxed{\frac{1}{2^{n/2}} \sum_{X=1}^{2^n-1} |X, f(X)\rangle^{n+1}}$$

$$|X, f(X)\rangle^{n+1} = |X\rangle^n \otimes |f(X)\rangle$$

contains all $2^n$ values of $f(X)$, but any measurement of the output state will project it into a single basis state...

thus, a single measurement can retrieve $f(X)$ for just one value of $X$. We would need $2^n$ measurements to map $f(X)$ entirely!

*We need algorithms that can extract the information "hidden" in the superposition of states.*

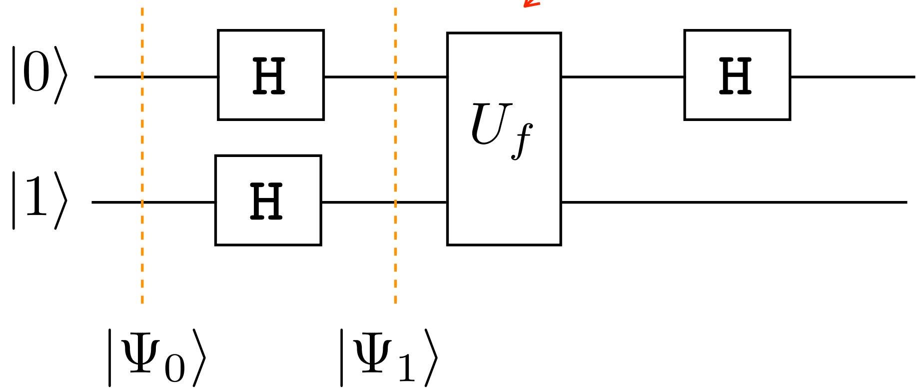
# The Deutsch's algorithm:

*it combines quantum parallelism and interference to compute two values of a given function, outperforming any classical algorithm.*

Let us assume $f: \{0,1\} \longrightarrow \{0,1\}$

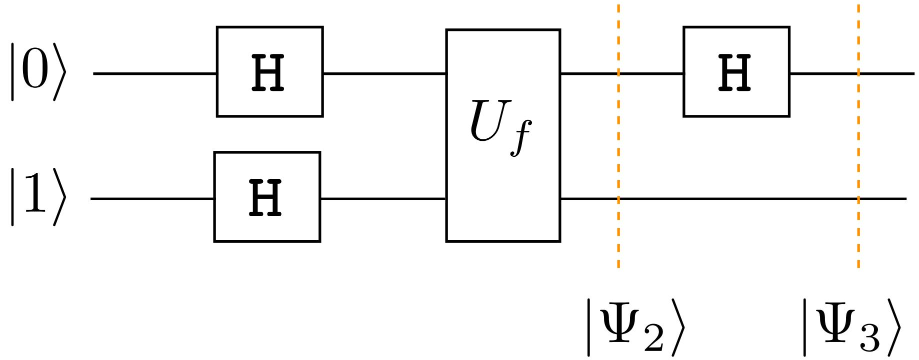
Consider the circuit

*function $f(x)$ evaluated only once*




$$|\Psi_0\rangle = |01\rangle$$

$$|\Psi_1\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}} \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}} = \frac{1}{2}\big(|00\rangle - |01\rangle + |10\rangle - |11\rangle\big)$$




$$|\Psi_2\rangle = \frac{1}{2}\Big(|0, f(0)\rangle - \big|0, \overline{f(0)}\big\rangle + |1, f(1)\rangle - \big|1, \overline{f(1)}\big\rangle\Big).$$

$$\boxed{\begin{aligned} 0 \oplus x &= x \\ 1 \oplus x &= \bar{x} \end{aligned}}$$

$$\begin{aligned}
|\Psi_3\rangle = \frac{1}{2}\Bigg(&\frac{|0, f(0)\rangle + |1, f(0)\rangle}{\sqrt{2}} - \frac{\big|0, \overline{f(0)}\big\rangle + \big|1, \overline{f(0)}\big\rangle}{\sqrt{2}} \\
&+ \frac{|0, f(1)\rangle - |1, f(1)\rangle}{\sqrt{2}} - \frac{\big|0, \overline{f(1)}\big\rangle - \big|1, \overline{f(1)}\big\rangle}{\sqrt{2}}\Bigg) \\
= \frac{1}{\sqrt{8}}\Big[&|0\rangle \otimes \Big(|f(0)\rangle + |f(1)\rangle - \big|\overline{f(0)}\big\rangle - \big|\overline{f(1)}\big\rangle\Big) \\
&+ \ |1\rangle \otimes \Big(|f(0)\rangle - |f(1)\rangle - \big|\overline{f(0)}\big\rangle + \big|\overline{f(1)}\big\rangle\Big)\Big].
\end{aligned}$$

Notice the following:

$$|\Psi_3\rangle = \frac{1}{\sqrt{8}}\Big[|0\rangle \otimes \Big(|f(0)\rangle + |f(1)\rangle - |\overline{f(0)}\rangle - |\overline{f(1)}\rangle\Big) + |1\rangle \otimes \Big(|f(0)\rangle - |f(1)\rangle - |\overline{f(0)}\rangle + |\overline{f(1)}\rangle\Big)\Big]$$

if $f(0) + f(1) = 0$
$$\begin{cases} f(0) = f(1) \\ |\Psi_3\rangle = \frac{1}{\sqrt{2}}|0\rangle \otimes \Big(|f(0)\rangle - |\overline{f(0)}\rangle\Big) \end{cases}$$

*will measure "0"*

if $f(0) + f(1) = 1$
$$\begin{cases} f(0) \neq f(1) \\ |\Psi_3\rangle = \frac{1}{\sqrt{2}}|1\rangle \otimes \big(|f(0)\rangle - |f(1)\rangle\big) \end{cases}$$

*will measure "1"*

*We can evaluate the sum $f(0)+f(1)$ without evaluating both $f(0)$ and $f(1)$.*
*We gained a factor of 2 in efficiency with respect to a classical algorithm.*

# The Deutsch - Jozsa algorithm: *(the n-qubit extension of Deutsch's algorithm)*

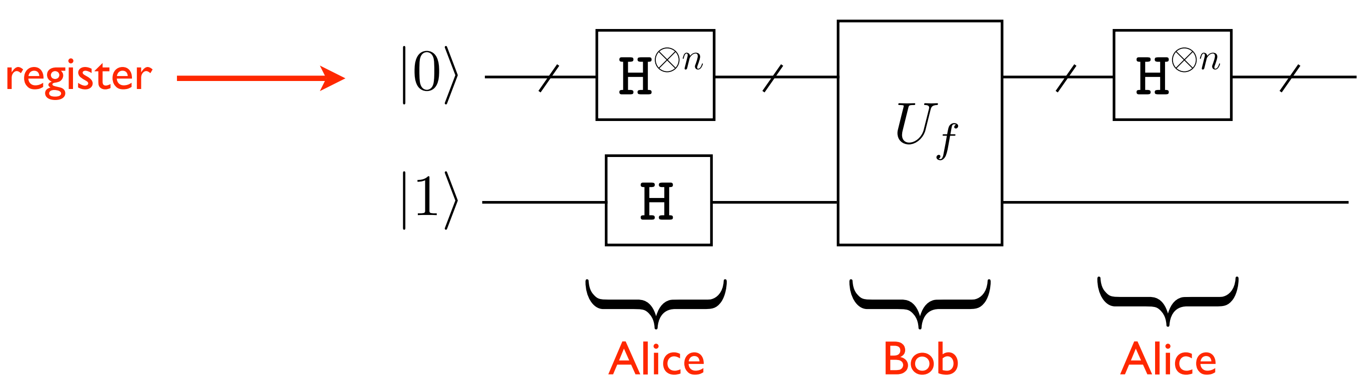
*Deutsch's problem:*

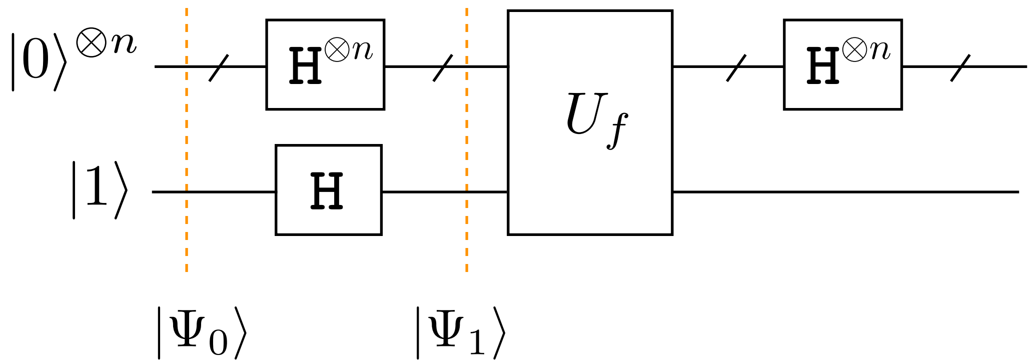
- Alice can send one number $X$ (between 0 and $2^n$ -1) to Bob each time.
- Bob uses $x$ to evaluate $f(X)$, $f : \{0,1\}^n \rightarrow \{0,1\}$ .
- The function $f(X)$ is either constant or balanced.
- How many times does Alice need to communicate with Bob to find out the nature of $f(X)$?

⇨ Classically, Alice would need to send $2^n/2+1$ numbers to Bob to determine $f(X)$ (worst case scenario - all first $2^n/2$ evaluations gave the same result). In the best case she would need to send at least 2.

⇨ Using qubits and unitary operations, Alice would need to send just 1 number to Bob to determine $f(X)$, thanks to quantum parallelism and interference!

## the circuit





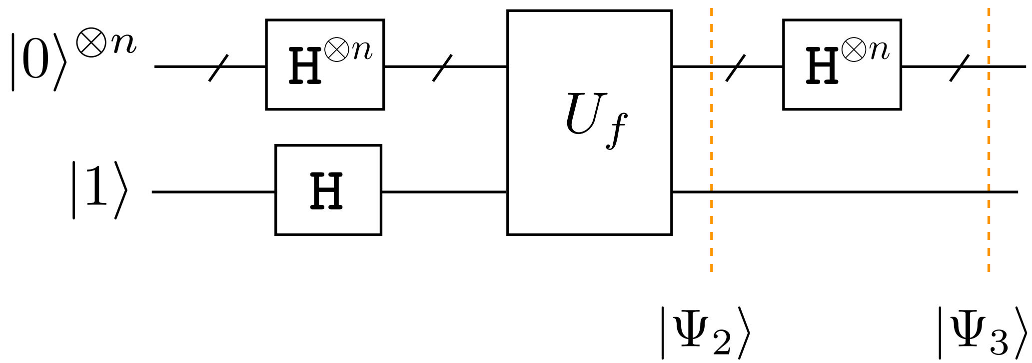

$$|\Psi_0\rangle = |0\rangle^{\otimes n} \otimes |1\rangle$$

$$\begin{cases} H^{\otimes n}|0\rangle^{\otimes n} = \left(H\,|0\rangle\right)^{\otimes n} = \dfrac{1}{2^{n/2}} \displaystyle\sum_{x_1, x_2, \ldots, x_n} |x_1 x_2 \cdots x_n\rangle \\ H^{\otimes n}|1\rangle^{\otimes n} = \dfrac{1}{2^{n/2}} \displaystyle\sum_{x_1, x_2, \ldots, x_n} (-1)^{x_1 \oplus x_2 \oplus \cdots \oplus x_b} |x_1 x_2 \cdots x_n\rangle \end{cases}$$

exercise: show it!

$$|\Psi_1\rangle = \frac{1}{2^{(n+1)/2}} \sum_{x_1, x_2, \ldots, x_n} |x_1 x_2 \cdots x_n\rangle \otimes \big(|0\rangle - |1\rangle\big)$$

*notice that:* $$H^{\otimes n}|X\rangle^{\otimes n} = \frac{1}{2^{n/2}} \sum_{z_1, z_2, \ldots, z_n} (-1)^{z_1 x_1 \oplus z_2 x_2 \oplus \cdots \oplus z_n x_b} |z_1 z_2 \cdots z_n\rangle$$




$$
\begin{aligned}
U_f\left[|X\rangle^n \otimes \big(|0\rangle - |1\rangle\big)\right] &= |X\rangle^n \otimes |0 \oplus f(X)\rangle - |X\rangle^n \otimes |1 \oplus f(X)\rangle \\
&= |X\rangle^n \otimes |f(X)\rangle - |X\rangle^n \otimes \left|\overline{f(X)}\right\rangle \\
&= (-1)^{f(X)}\,|X\rangle^n \otimes \big(|0\rangle - |1\rangle\big)
\end{aligned}
$$

$$
|\Psi_2\rangle = \frac{1}{2^{(n+1)/2}} \sum_{X=0}^{2^n-1} (-1)^{f(X)}\,|X\rangle^n \otimes \big(|0\rangle - |1\rangle\big)
$$

$$
\begin{aligned}
|\Psi_3\rangle &= H^{\otimes n}|\Psi_2\rangle \\
&= \frac{1}{2^{n+1/2}} \sum_{X,Z=0}^{2^n-1} (-1)^{f(X)\oplus X\cdot Z}|Z\rangle^n \otimes \big(|0\rangle - |1\rangle\big)
\end{aligned}
$$

$$(Z\cdot X = z_1x_1 + z_2x_2 + \cdots z_nx_n)$$

$$|\Psi_3\rangle = \sum_{X,Z=0}^{2^n-1} \underbrace{\frac{(-1)^{f(X)\oplus X\cdot Z}}{2^n}}_{A(X,Z)} |Z\rangle^n \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}}$$

*constant f(X):*

$(Z=0)$

$$\sum_{X=0}^{2^n-1} A(X,0) = \sum_{X=0}^{2^n-1} \frac{(-1)^{f(X)}}{2^n} = \pm 1$$

*But* $|\Psi_3| = 1 \longrightarrow |\Psi_3\rangle = |0\rangle^{\otimes n} \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}}$

*balanced f(X):*

$(Z=0)$

$$\sum_{X=0}^{2^n-1} A(X,0) = \sum_{X=0}^{2^n-1} \frac{(-1)^{f(X)}}{2^n} = \frac{(-1)^0 + (-1)^1}{2} = 0$$

*The state* $|0\rangle^{\otimes n}$ *has zero amplitude* $\longrightarrow$ *at least one qubit will be* $|1\rangle$

*If Alice measures at least one qubit in the state "1", the function is balanced; otherwise it is constant.*

Q: Are there quantum algorithms that are superior to classical ones *and* useful at the same time?

A: Yes!

*Three classes are known:*

- Algorithms based on Fourier transform (e.g. Shor's factorization algorithm).

- Search algorithms (e.g. Grove's).

- Simulation of quantum systems.

We will see how they work in this course.

# 2.1 Classical Computation

computation: (abstract) — *input* (symbols) $\xrightarrow{\textit{algorithm}}$ *output* (symbols)

Ex: *input:* **a,b** $\longrightarrow$ *output:* **a x b = c**

you learn the algorithm in elementary school

**a**,**b**,**c** represented by integers (base 2, 10, ...)

computation: (physical) — *input* (objects,states) $\xrightarrow{\textit{physical process}}$ *output* (objects,states)

Ex: *input, output:* resistors, capacitors, and transistors (voltages, currents)

*algorithm:* sequence of current and voltage pulses

Q: When is a function computable?
A: When an algorithm for its evaluation exists!

Q: Are all functions computable? ⟷ Is there an algorithm that can solve all mathematical problems? (Hilbert)
A: No!

Q: How do we know that?
A: Gödel, Turing, ...

*There are conjectures that cannot be proved.*

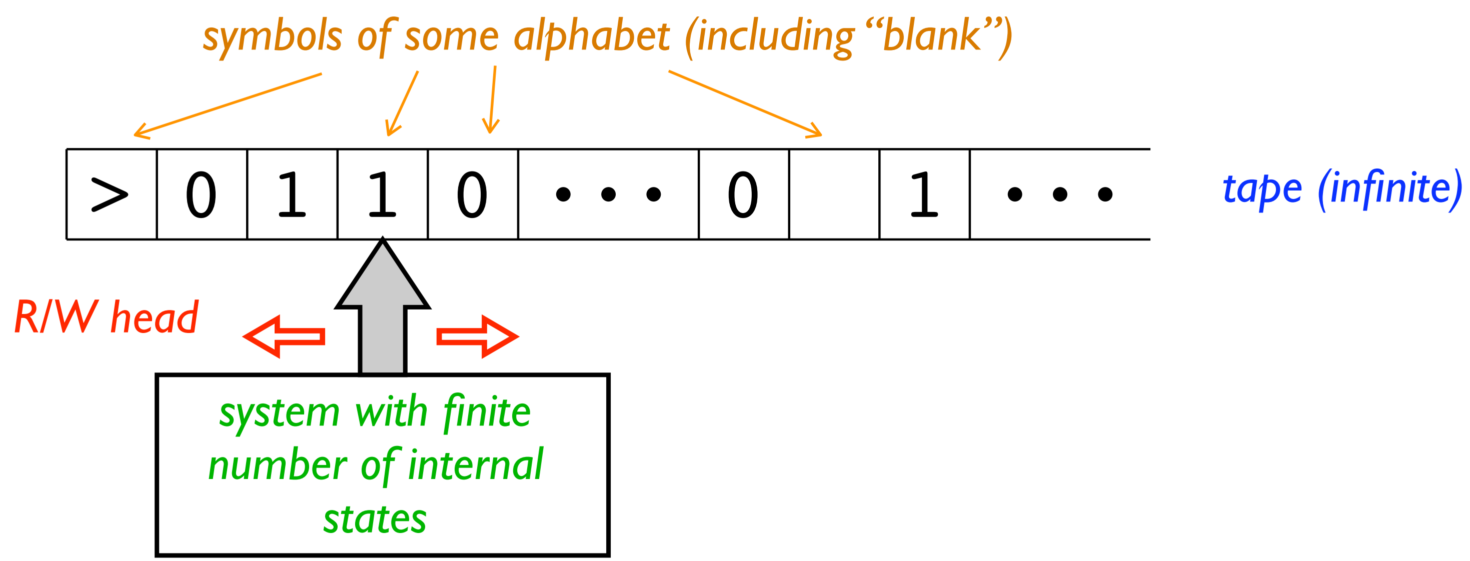
*There are functions that cannot be computed algorithmically.*

Church-Turing thesis: Any function that is computable by any means is computable by a "Turing machine".

*Turing machines are model computers (not necessarily efficient ones...).*

2C

# Turing machines




"halt"

Finite set of machine states: $Q = \{q_0, q_1, \ldots, q_n; q_h\}$

Symbols of the alphabet: $S = \{\ \ , >, 0, 1, \ldots\}$

Direction of head movement: $d \in \{L, O, R\}$

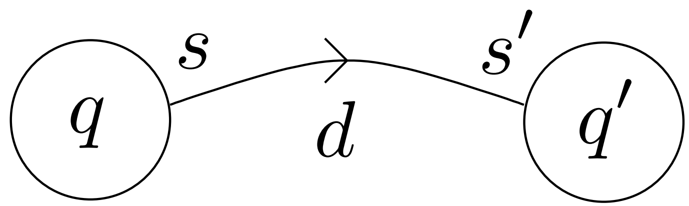
The machine operates in steps: $(q, s) \longrightarrow (q', s', d)$ *quintuple*

The finite-state system contains the "code".

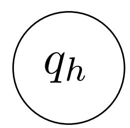
The tape contains the "input" and "output" data.

The machine halts when it encounters the "halt" state.

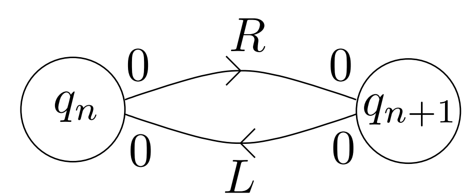
$(q, s, q', s', d)$



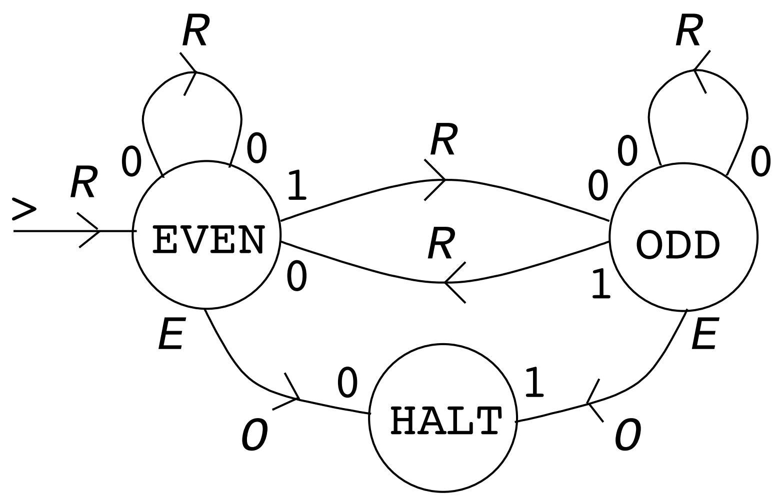

**Ex:** halting state $(q_h, s, q_h, s, O)$




**Ex:** infinite loop $(q_n, 0, q_{n+1}, 0, R)$ $(q_{n+1}, 0, q_n, 0, L)$



# Ex: a parity counter machine




| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| EVEN | $\boxed{>}$ | 0 | 0 | 1 | 1 | 0 | 1 | *E* |
| EVEN | > | $\boxed{0}$ | 0 | 1 | 1 | 0 | 1 | *E* |
| EVEN | > | 0 | $\boxed{0}$ | 1 | 1 | 0 | 1 | *E* |
| EVEN | > | 0 | 0 | $\boxed{1}$ | 1 | 0 | 1 | *E* |
| ODD | > | 0 | 0 | 0 | $\boxed{1}$ | 0 | 1 | *E* |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| EVEN | > | 0 | 0 | 0 | 0 | $\boxed{0}$ | 1 | *E* |
| EVEN | > | 0 | 0 | 0 | 0 | 0 | $\boxed{1}$ | *E* |
| ODD | > | 0 | 0 | 0 | 0 | 0 | 0 | $\boxed{E}$ |
| HALT | > | 0 | 0 | 0 | 0 | 0 | 0 | $\boxed{1}$ |

result of the computation

2F

# Universal Turing Machines and the Halting Problem

*It is possible to label or tag a Turing machine uniquely? Yes!*

*TM:*

$$Q \times S \longrightarrow Q \times S \times D$$
$$(q, s) \longrightarrow (q', s', d)$$

⟵ function that takes a discrete set into another (countable)

$$f: \{1, 2, \ldots, N\} \longrightarrow \mathcal{N}$$
$$x \longrightarrow f(x)$$

⟵ can be labeled by the integer number

$$n_f = \prod_{i=1}^{N} p_i^{f_i}$$

$p_i$ is the $i$th prime

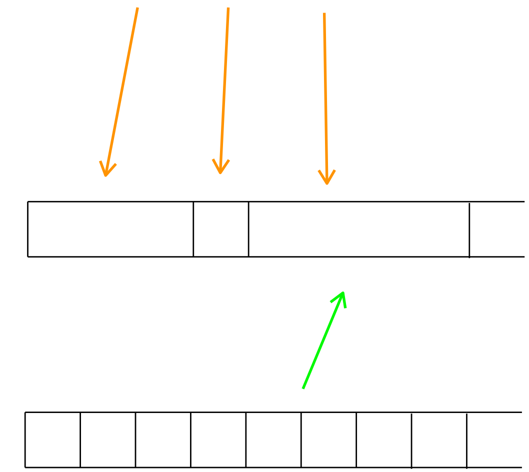
*The universal Turing machine (UTM) simulates any TM*

Input in tape: Turing number $m$ + “blank” + data

| | | | |
|---|---|---|---|
| $m$ | | $x$ | |

# A Universal Turing Machine simulates any Turing Machine

$$TM\,(x) \;=\; UTM\,(m\,//\quad//\,x\,)$$



⇨ UTM: *a "programmable" computer!*

It is possible to build an UTM with 8 symbols and 23 states (Minsky, 1967).

can be reduced to 6 (Feynman)

There are ways to build an UTM with 2 states and lots of symbols, or vice-versa!

## Generalizations of TM

- Two or more tapes (not necessary, but useful):

$$M: \ (q, s_1, s_2) \longrightarrow (q', s_1', s_2', d_1, d_2)$$

- Probabilistic TM:

$$(q, s) \longrightarrow \begin{array}{ll} (q_1', s_1', d_1) & \textit{with probability } p_1 \\ \quad \vdots & \\ (q_n', s_n', d_n) & \text{``} \qquad \text{``} \qquad p_n \end{array} \qquad \sum_{i=1}^{n} p_i = 1$$

$$\Downarrow$$

It allows for *randomized algorithms* (sometimes very powerful ones).

Revised Church-Turing thesis: *Any algorithmic process can be simulated efficiently using a probabilistic Turing machine.*

Halting problem

Q: Can we predict if a machine will halt for a given input?

A: Not in general!

Let $T_F$ (x) be a TM that outputs $f(x)$ when fed with $x$.

Let D(t,x) be a UTM that takes $T_F$ and $x$ as input.

*assumption: D always halts.*

If $T_F$ halts with $x$, D yields 1; otherwise, it yields 0.

Let Z be another UTM such that:

- if $T_F$ $(x)$ halts, i.e., $D(t,x) = 1$, $Z(d)$ does not halt (infinite loop);
- if $T_F$ $(x)$ does not halt, i.e., $D(t,x) = 0$, $Z(d)$ halts and yields 0.

Substitute $T_F$ by Z: $D(z,x)$; arrive at a contradiction!

Z halts if and only if Z does not halt.

$\Rightarrow$ $\boxed{\text{D cannot exist.}}$

Q: How many other uncomputable functions are out there?

A: Many! Certainly more than computable ones...

Each computable function can be related to a Turing Machine.

Turing Machines are *countable* (each one represented by an integer number).

Computable functions are countable, but uncomputable functions are not.

*(It is a bit like the difference between real and integer or rational numbers.)*

But even if the function is computable, it may not mean much:

Algorithms need not just be effective, but also efficient to be practical.

The emphasis is not so much on whether a function is computable or not, but on whether an efficient algorithm exists for it.

*What is an efficient algorithm?*

# 2.2 Computation Complexity

*How much time and memory does it take to solve a computational problem?*

Computational complexity: look for *lower bounds* (best possible algorithm).

Two main classes of problems: for an input consisting of a $n$-bit number,

(i) resources ~ $O(n^a)$ Polynomial problems (easy, fast, tractable);

(ii) resources ~ $O(e^n)$ Exponential problems (hard, slow, intractable).

Ex: compute the sum of two binary numbers, $x_1 x_2 \dots x_{m_1}$ and $y_1 y_2 \cdots y_{m_2}$ .

⇨ number of elementary operations ~ $O(n)$, $n = m_1 + m_2$ .

Ex: find the prime factors of an binary integer $x_1 x_2 \dots x_n$ .

⇨ *believed* to be an intractable (exponential) problem!

*Why? Take for instance the "dumb" method:* $\left\{\begin{array}{l} \text{given } N\text{, take } \sqrt{N} \\ \text{for } i = 2 \text{ to } [\sqrt{N}] \\ \quad \text{check if } N/i \in \mathcal{N} \end{array}\right.$

$\Rightarrow$ number of elementary steps ~ $\sqrt{N}$

$n$ bit number, $N \sim 2^n$

number of steps ~ $2^{n/2} \sim e^{(\frac{\ln 2}{2})\, n}$ $\longleftarrow$ exponential in $n$

Is there a *O(n* $^{a}$*)* algorithm? Nobody knows for sure.

Word of cautious: $n^{1000}$ vs. $e^n$ $\Rightarrow$ exponential is "better" if $n < 9500$

Q: Is the definition of complexity class computer dependent?

A: No.

Church-Turing (strong) thesis: A probabilistic TM can simulate any model of computation with an overhead at most of polynomial order.

# 2.3 Complexity Classes

Best way to study complexity classes is through *decision problems*.

("yes" or "no" answer)

Technical approach: *Formal languages*

$\Sigma \longrightarrow$ alphabet set

$\Sigma^* \longrightarrow$ set of finite strings of symbols in $\Sigma$

$L \longrightarrow$ subset of $\Sigma^*$

Ex: $\Sigma = \{0, 1, \ldots, 9\}$ $\quad \Sigma^* = \{0, 1, \ldots, 9\, 10, 11, \ldots, 99, 100, \ldots\}$

$L_1 = \{0, 2, 4, 6, \ldots\}$ $\quad L_2 = \{1, 2, 3, 5, 7, 11, \ldots\}$

*even numbers* *prime numbers*

Or, use binary representation: $\Sigma = \{0, 1\}$ $\quad \Sigma^* = \{0, 1, 10, 11, 100, 101, \ldots\}$

The problem: Given an $x \in \Sigma^*$, is $x \in L$ ?

The solution: input $x$ in a TM and wait for a "yes" or a "no".

complexity class P $\Rightarrow$ problems with $O(n^a)$ solution

> A problem is in P if the TM can decide if $x \in L$ in a time that scales as a power of the length of $x$ .

Q: Are there problems not in P ?

A: Yes! But hard to prove... Conjectures abound.

$\downarrow$

*"Given a composite integer m and another integer l, l<m, does m have a non-trivial factor less than l? "*

(believed to be outside P)

## complexity class NP

⇨ problems that can be checked easily (polynomial time) if a guess ("witness") is provided:

*'yes'* or *'no'* instances are decided in polynomial time for a given attempted witness.

Q1: Does 347 have a factor smaller than 17? *hard* ⟵ NP problem

Q2: Is 7 a factor of 347? *easy*

7 is the attempted witness.

The most famous problem of Computer Science

$\boxed{\text{Is } P \neq NP \text{ ?}}$

The answer is believed to be *'yes'*...

Ex: Given $H$, does $H$ have an eigenvalue $E<E^*$?

*Who could play the role of a witness?*

# Two important concepts:

## (i) Reducibility

P$_1$ : $x \in L$ ?

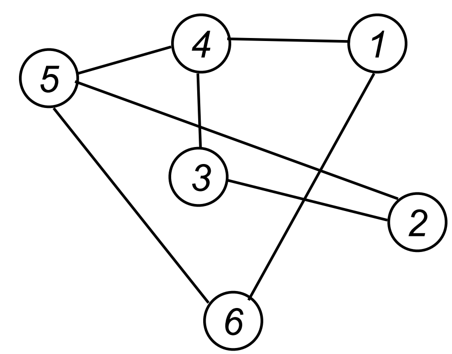
P$_2$ : $x \in L'$ ?

$L'$ is reducible to $L$ if there is function $R(x)$ computable in polynomial time such that $x \in L' \leftrightarrow R(x) \in L$ .

Ex:

1) Does a graph have a Hamiltonian cycle (HC)?

(A Hamilton cycle passes through all vertices once.)




2) Given a city distance matrix $d_{ij},\ i,j = 1,\ldots,n$ is there a way to tour all cities with distance less than $d$ (TSP)?

*Prove* $\boxed{\text{HC} \longleftrightarrow \text{TSP}}$

*If an algorithm exists for solving HC, it can also be used to solve TSP (with little additional work).*

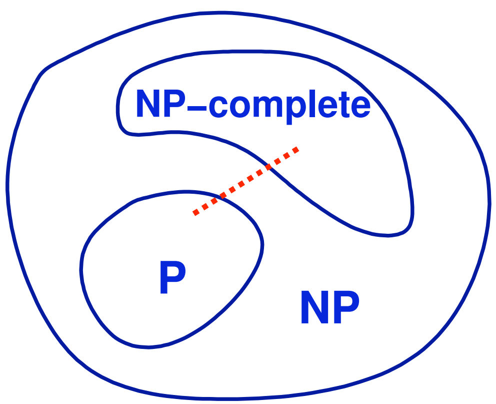
HC and TSP are in the same class.

(ii) Complete problem in a class: If we can decide about $x \in L$, we can decide for all other $L'$ in the same class.

⇨ Any problem in P is complete. *(all problems in P are equally hard)*

<u>NP-complete problems:</u>

If we solve one problem in NP-complete, we could solve any problem in NP (with at most a polynomial overhead).




*theorem* If NP-complete $\cap$ P $\neq \emptyset$, then NP=P.

*(prove it!)*

# 2.4 Energy and Dissipation in Computation

Q: What is the minimum energy required to do computation?

A: In principle, it can be zero!

*Energy consumption is related to reversibility of computation.*

Landauer principle (1961): The entropy of the environment increases by at least $k_B \ln 2$ when a single bit of information is erased.

Recall Boltzmann definition of entropy: $S = k_B \ln \Gamma$ ← number of accessible states

↑ Boltzmann constant (1.38 x $10^{-23}$ J/K)

Each bit: $S = k_B \ln 2$

Erase a bit: $\Delta S = k_B \ln 2$ ← entropy lost to the environment

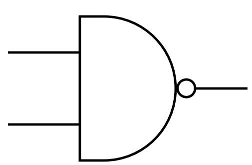
*(Ideal operation, "isolated" bit.)*

Free energy variation: $\Delta F = T\,\Delta S$ $\Rightarrow$ $\boxed{\Delta F = k_B T \ln 2}$

*minimum energy required to erase binary information irreversibly*

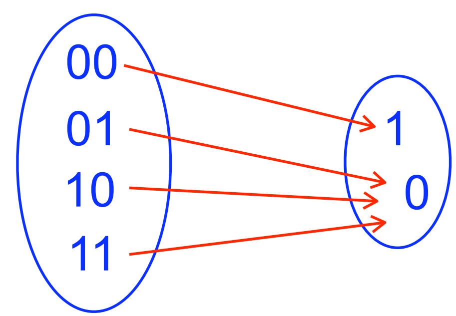
<u>In some (non rigorous) sense:</u> *Loss of information ≈ loss of entropy ≈ dissipation*

Ex: NAND gate



2 bits → 1 bit

| $a$ | $b$ | $c$ |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 0 |




*if the output is 0, we cannot reverse the operation*

1 bit is lost (erased)

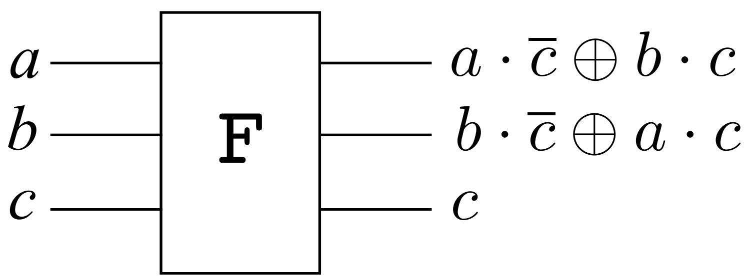
information is lost

irreversible operation, energy cost $\Delta E \geq k_B T \ln 2$

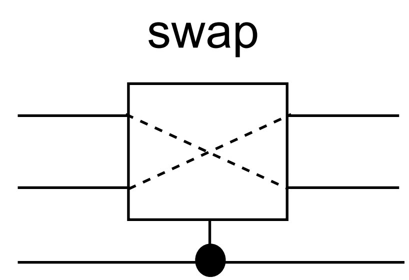
Q: How can we carry out logical operations without spending energy?

A: Use *reversible* gates!

## The Fredkin gate (1982)



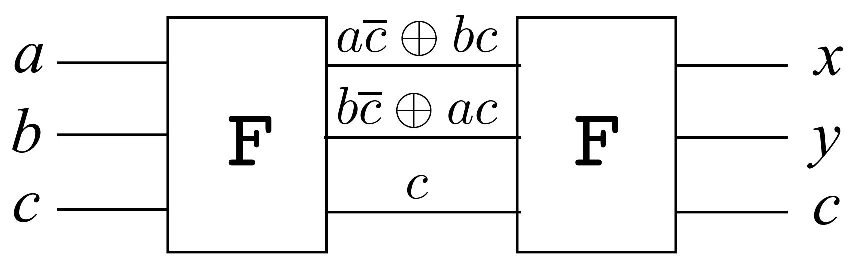

$\equiv$




$$\begin{cases} \text{if } c = 0, \text{ do nothing} \\ \text{if } c = 1, \text{ swap } a, b \end{cases}$$

$$c^2 = c$$
$$c\bar{c} = 0$$
$$c \oplus \bar{c} = 1$$
$$(a \oplus b)c = ac \oplus bc$$

*Reversible*



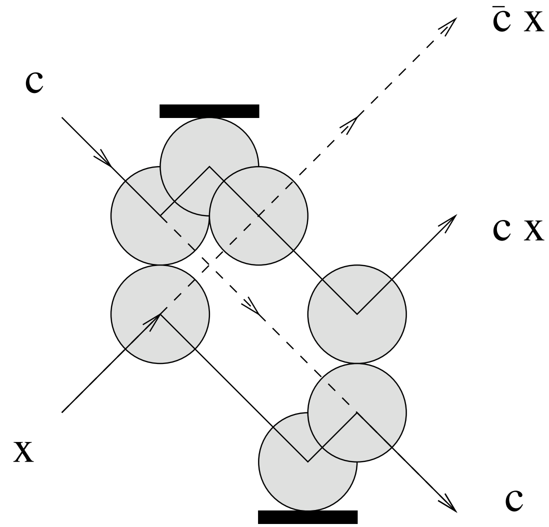

$$x = (a\bar{c} \oplus bc)\,\bar{c} \oplus (b\bar{c} \oplus ac)\,c$$
$$x = a\bar{c} \oplus ac$$
$$x = a$$

(somewhat obvious result... but check for $y$)

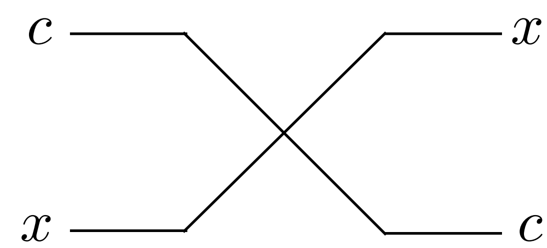
3D

$c$

Reversible gates can be implemented with billiard balls:




$\equiv$

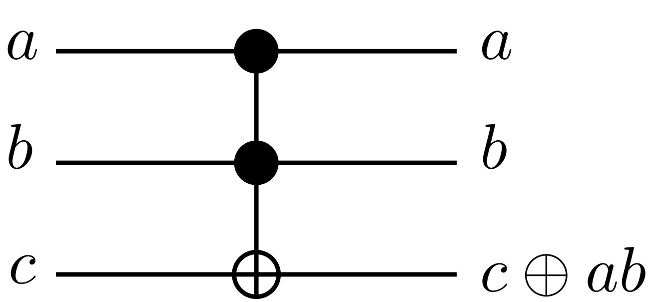



*a swap gate*

No surprise: Classical Mechanics is reversible!

*(In fact, all laws of Physics are time-reversal invariant.)*

<u>The Toffoli gate (1982)</u>



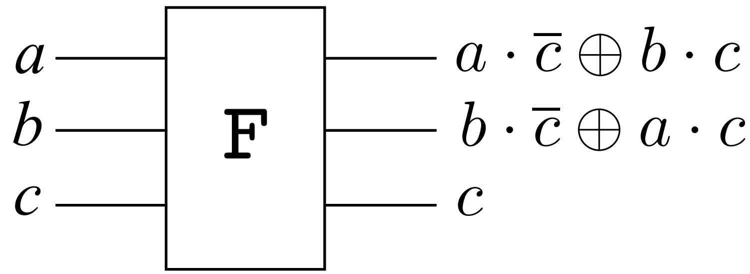

*It is its own inverse*

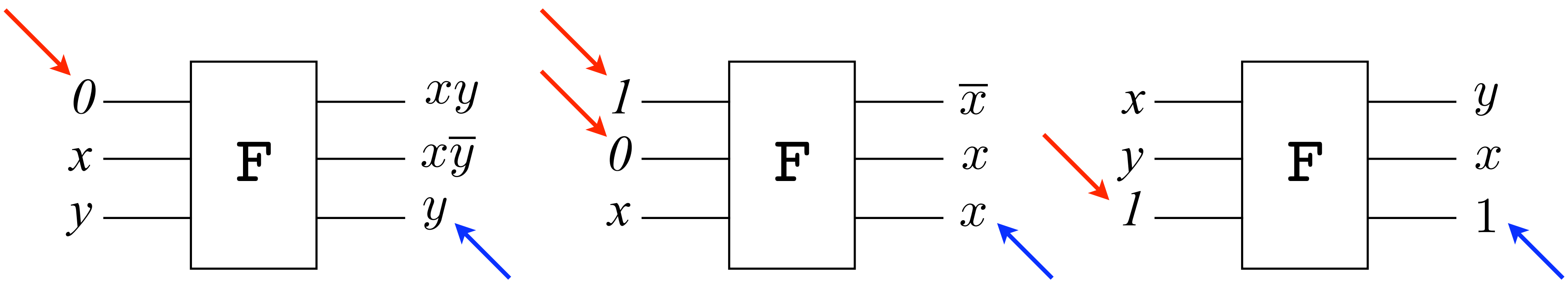
$$x = (c \oplus ab) \oplus ab$$
$$x = c \oplus (ab \oplus ab)$$
$$x = c$$

$$d \oplus d = 0 \bmod 2$$

Both Fredkin and Toffoli gates are *universal*:

*Fredkin*



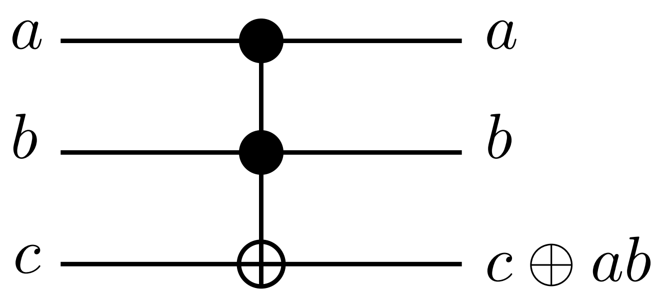

AND NOT,FANOUT CROSSOVER,SWAP

*ancilla bits*

*garbage bits*

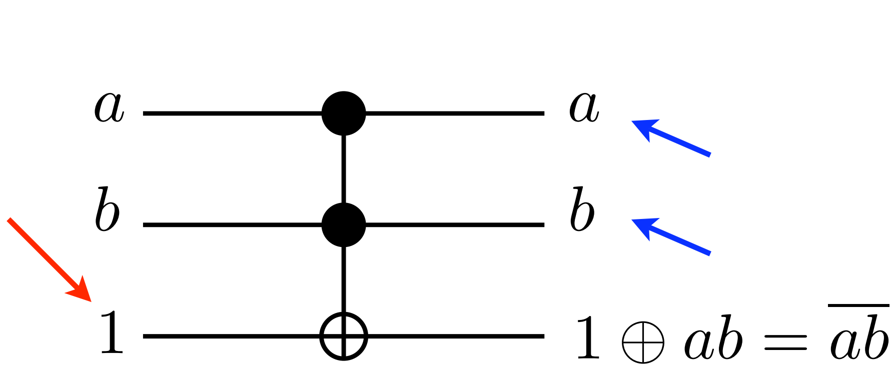
} price to pay for doing reversible computation

## *Toffoli*

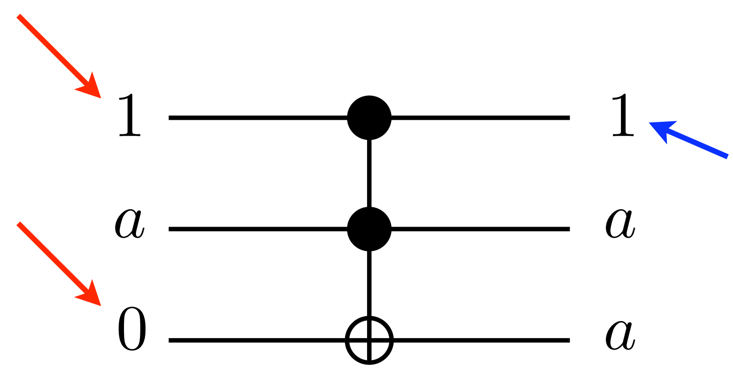



*ancilla bits*

*garbage bits*




NAND




FANOUT

OBS: It turns out that the Toffoli gate is also useful for Quantum Computation.

# Caveats of reversible computation:

1) Large susceptibility to noise ⇨ Requires error correction

*usually requires erasing bits*
*at the end (to save memory)...*

2) Additional overhead cost ⇨ extra bits;
preparation of ancilla bits;
dealing with garbage.

↑

*serious problem to Quantum Computation*

(can be fixed by a proper use of `NOT` and `C-NOT` gates)