Quantum Computing

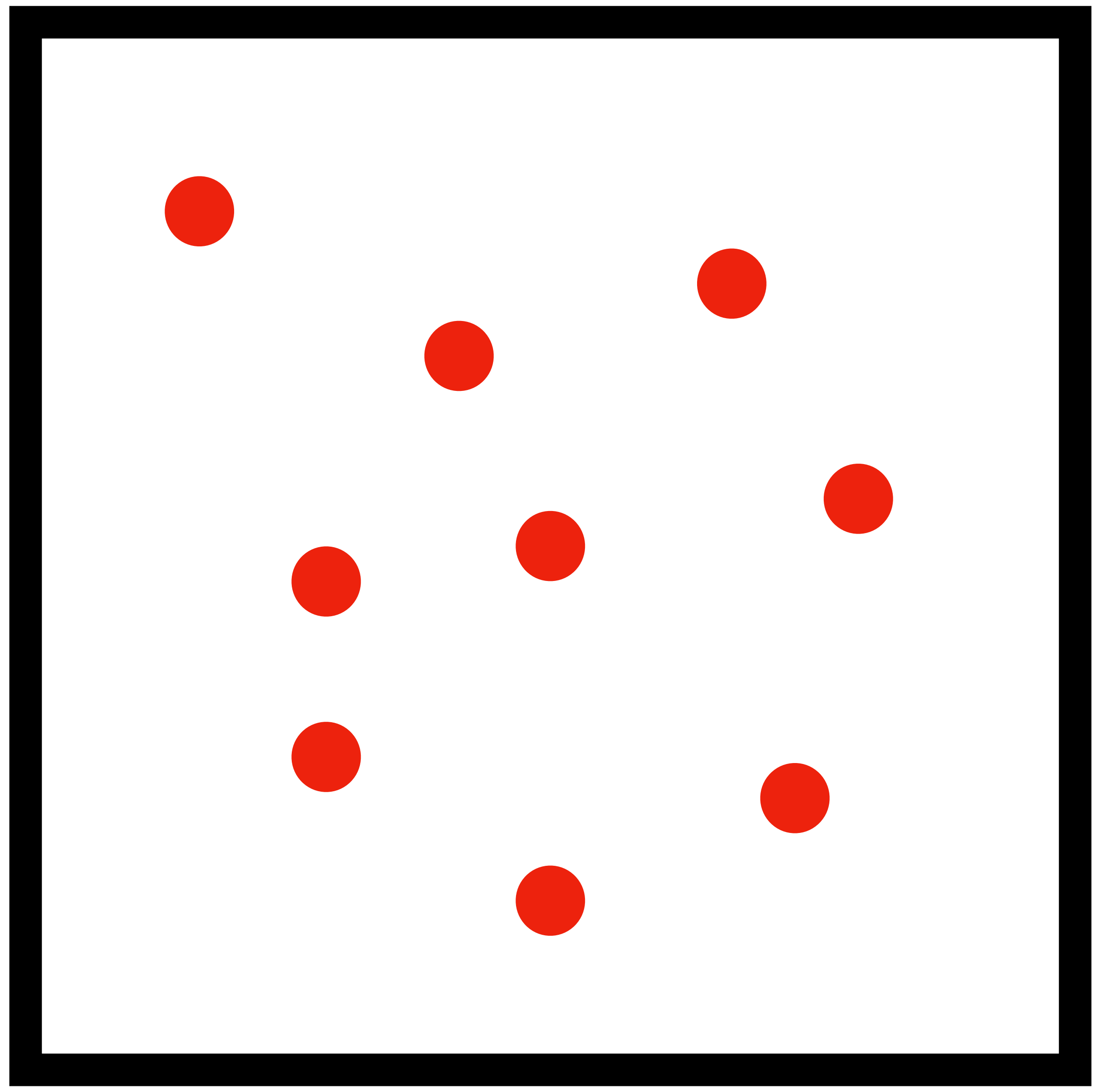
Algorithms and Complexity

Hakim Rachidi, Göttingen SoSe 25

Motivation

Complexity of Quantum

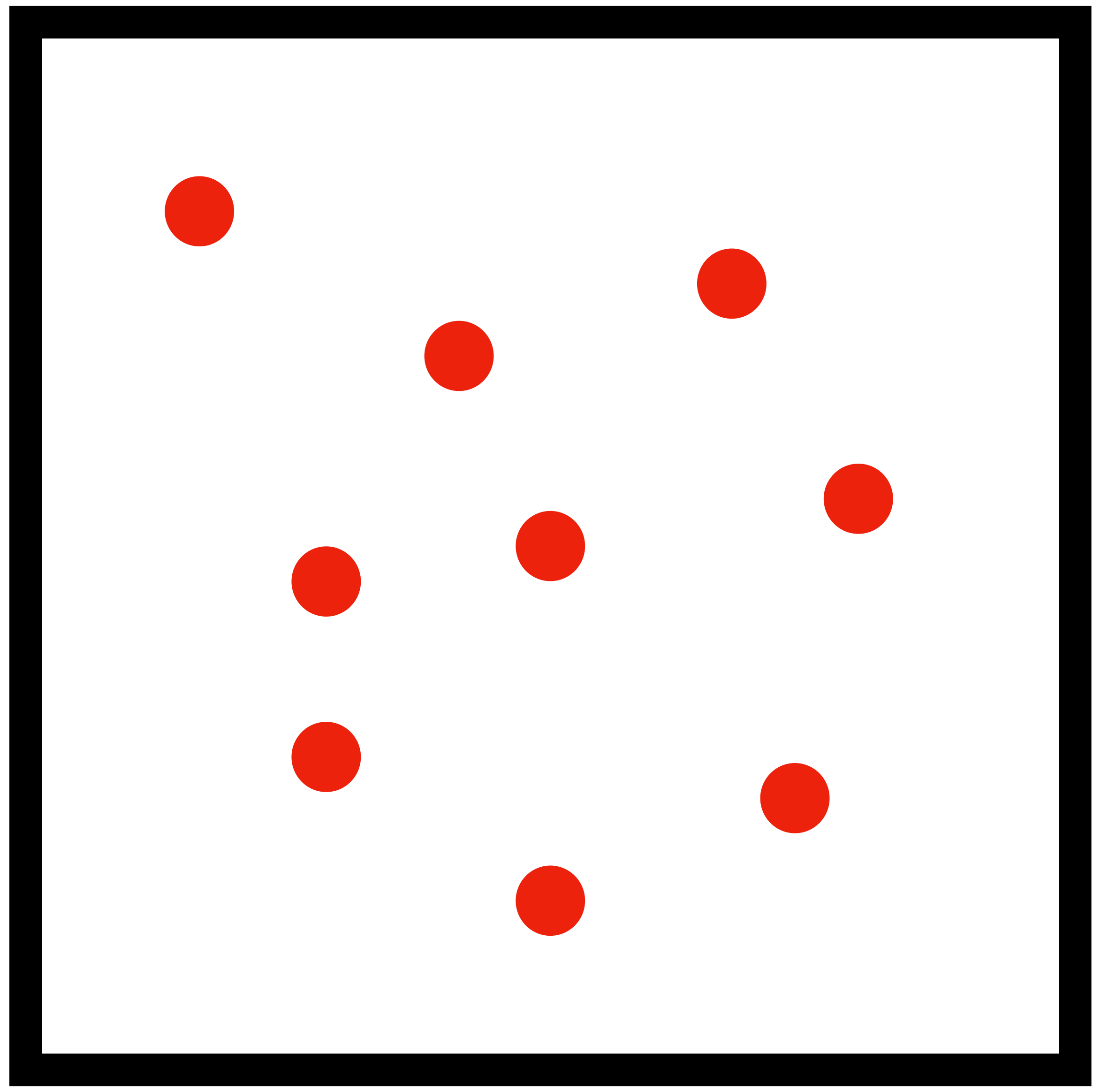
Classical



Motivation

Complexity of Quantum

Classical



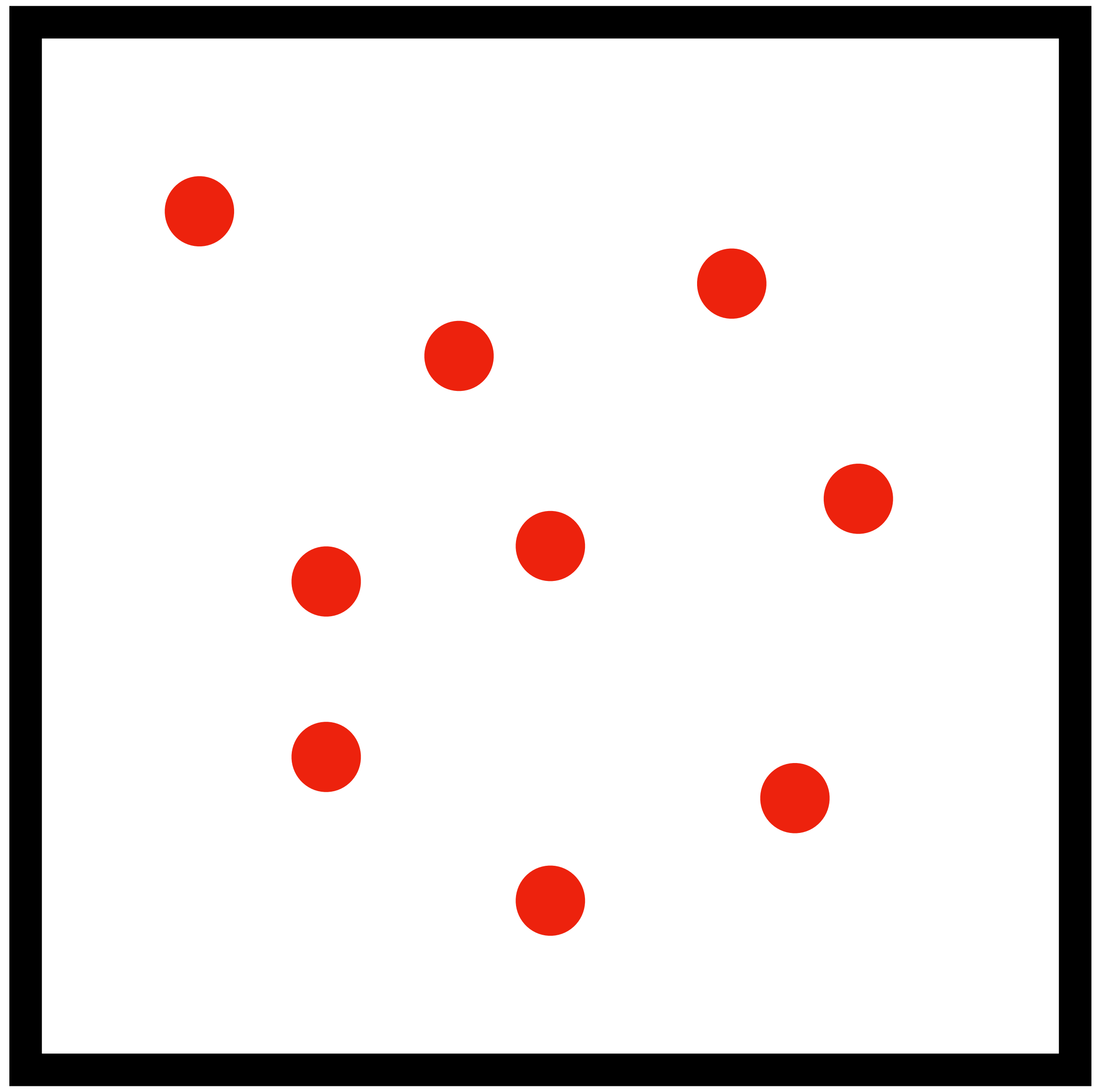
Position $q_i \in \mathbb{R}^f$

Momentum $p_i \in \mathbb{R}^f$

Motivation

Complexity of Quantum

Classical



Position $q_i \in \mathbb{R}^f$

Momentum $p_i \in \mathbb{R}^f$

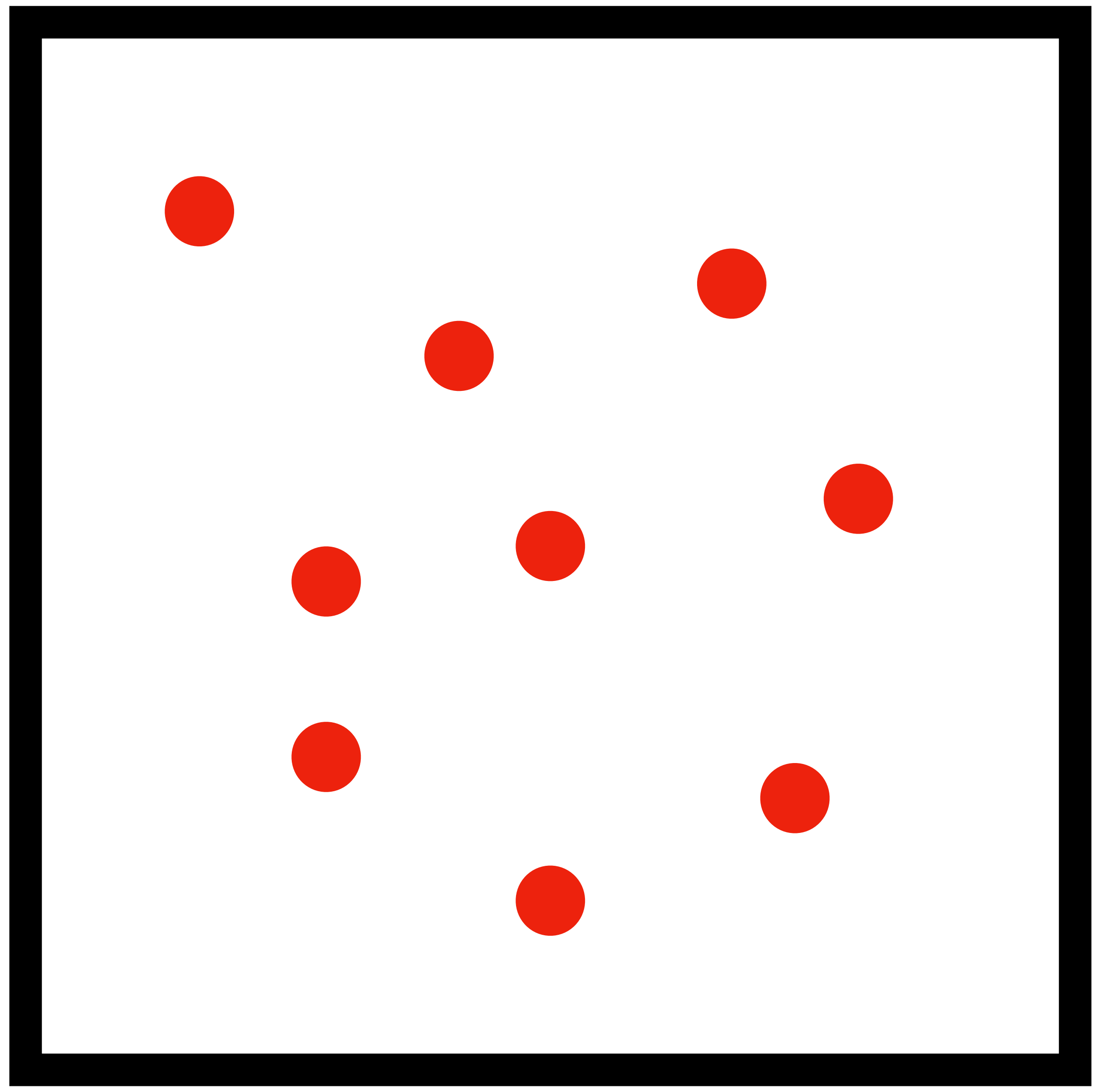
Phase space Γ consists of $(q_1, p_1, \ldots, q_N, p_N)$

$$\boxed{\dim \Gamma = 2fN}$$

Motivation

Complexity of Quantum

Classical



Position $q_i \in \mathbb{R}^f$

Momentum $p_i \in \mathbb{R}^f$

Phase space Γ consists of $(q_1, p_1, \ldots, q_N, p_N)$

$$\boxed{\dim \Gamma = 2fN}$$

Quantum

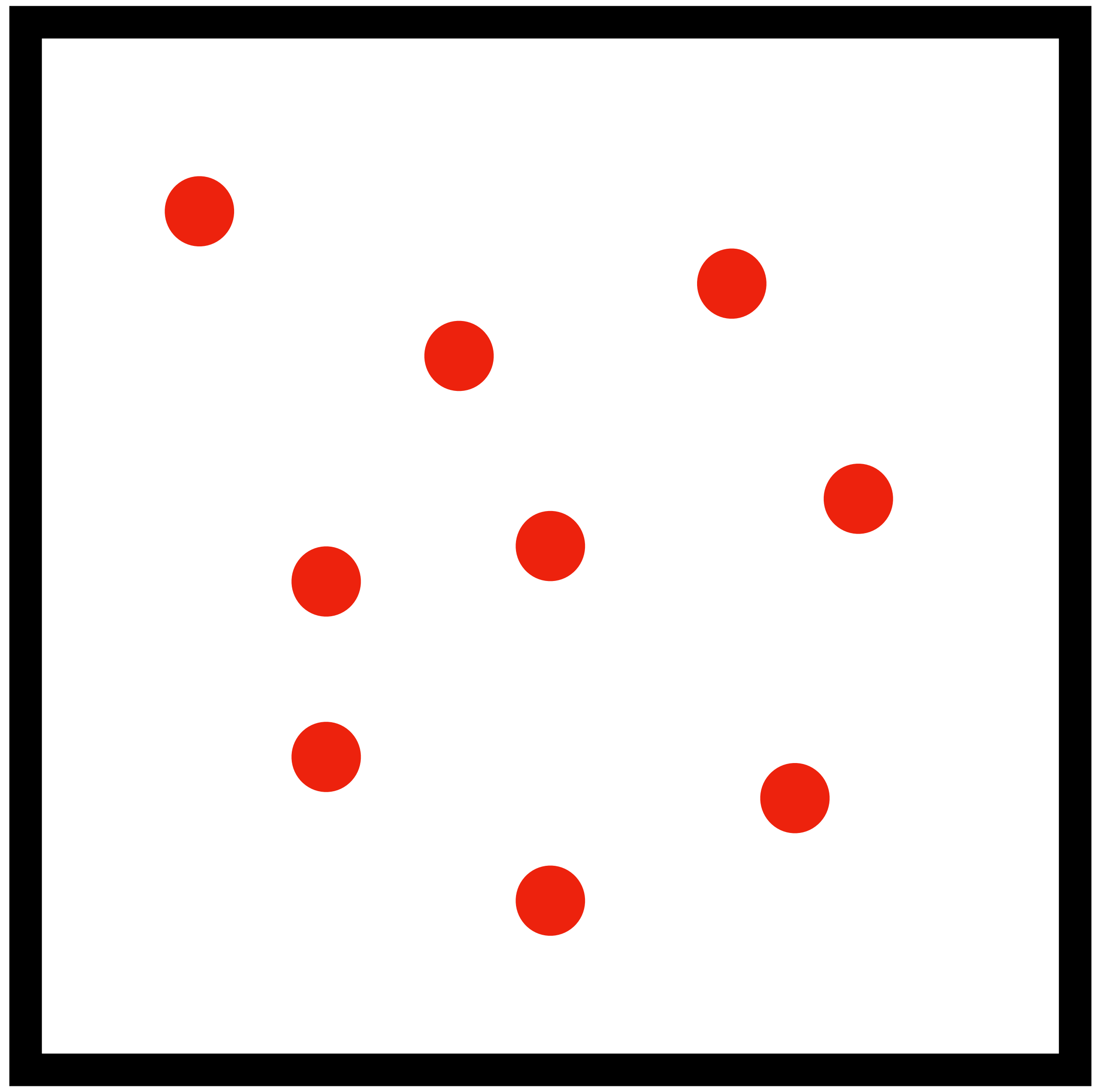
Ising chain

$|\psi_1\rangle\,|\psi_2\rangle\,|\psi_3\rangle\,|\psi_4\rangle\,|\psi_5\rangle$

Motivation

Complexity of Quantum

Classical



Position $q_i \in \mathbb{R}^f$

Momentum $p_i \in \mathbb{R}^f$

Phase space Γ consists of $(q_1, p_1, \ldots, q_N, p_N)$

$$\boxed{\dim \Gamma = 2fN}$$

Quantum

Ising chain

$|\psi_1\rangle\, |\psi_2\rangle\, |\psi_3\rangle\, |\psi_4\rangle\, |\psi_5\rangle$

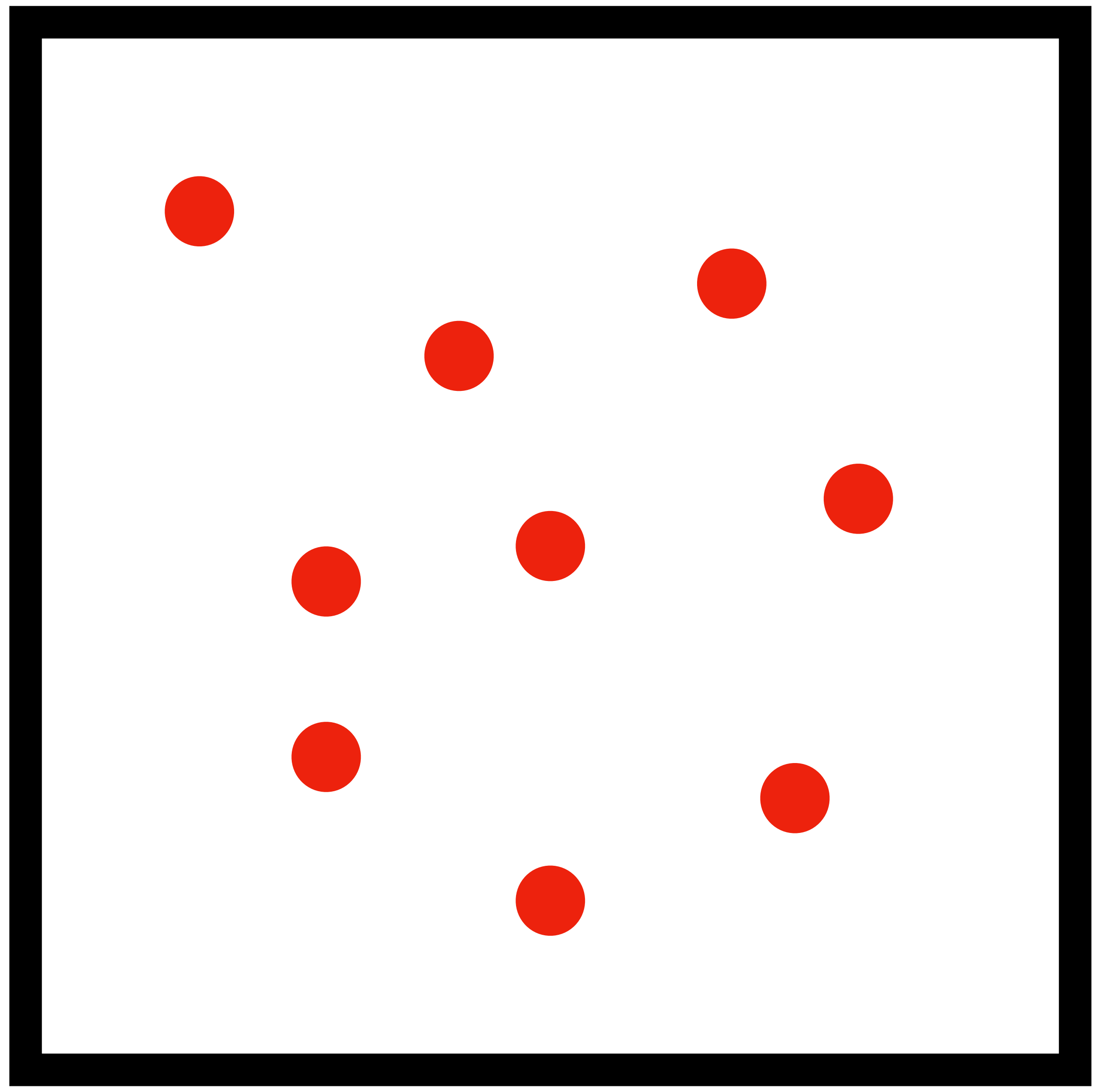
$$|\psi_i\rangle = \alpha |\uparrow\rangle + \beta |\downarrow\rangle \in \mathcal{H}_i$$

$$|\psi\rangle = |\psi_1\rangle \otimes \ldots \otimes |\psi_N\rangle \in \mathcal{H}$$

Motivation

Complexity of Quantum

Classical



Position $q_i \in \mathbb{R}^f$

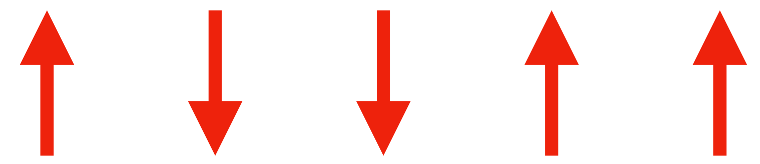
Momentum $p_i \in \mathbb{R}^f$

Phase space Γ consists of $(q_1, p_1, \ldots, q_N, p_N)$

$$\boxed{\dim \Gamma = 2fN}$$

Quantum

Ising chain



$|\psi_1\rangle\,|\psi_2\rangle\,|\psi_3\rangle\,|\psi_4\rangle\,|\psi_5\rangle$

$$|\psi_i\rangle = \alpha|\uparrow\rangle + \beta|\downarrow\rangle \in \mathscr{H}_i$$

$$|\psi\rangle = |\psi_1\rangle \otimes \ldots \otimes |\psi_N\rangle \in \mathscr{H}$$

$$\dim(\mathscr{H}_1 \otimes \mathscr{H}_2) = (\dim \mathscr{H}_1) \cdot (\dim \mathscr{H}_2)$$

$$\boxed{\dim \mathscr{H} = 2^N}$$

Motivation

Complexity of Quantum

Classical

$$\boxed{\dim \Gamma = 2fN}$$

Computational effort scales <u>polynomially</u>

Motivation

Complexity of Quantum

Classical

$$\boxed{\dim \Gamma = 2fN}$$

Computational effort scales
polynomially

Quantum

$$\boxed{\dim \mathscr{H} = 2^N}$$

for $N = 272$

$2^N >$ # Atoms in the visible universe

Computational effort scales
exponentially

Motivation
Complexity of Quantum

Classical

$$\boxed{\dim\Gamma = 2fN}$$

Computational effort scales <u>polynomially</u>

Feasible

Quantum

$$\boxed{\dim\mathscr{H} = 2^N}$$

for $N = 272$

$2^N >$ # Atoms in the visible universe

Computational effort scales <u>exponentially</u>

Infeasible

Motivation

Simulating quantum systems

Can we exploit quantum systems to simulate quantum systems?

— Feynmann (1982)

Motivation

Simulating quantum systems

Can we exploit quantum systems to

simulate quantum systems?

— Feynmann (1982)

→Universal Quantum Simulators (Quantum Computers)

Motivation

What is complex and what is efficient?

Multiplication

$$113 \cdot 73 = 8249$$

Number of steps linear in the number of digits

Motivation

What is complex and what is efficient?

Multiplication

$$113 \cdot 73 = 8249$$

Number of steps linear in the number of digits

Prime factoring

$$8249 = ?$$

Computational steps required grow exponentially in the number digits

Motivation

What is complex and what is efficient?

Multiplication

$$113 \cdot 73 = 8249$$

Number of steps linear in the number of digits

Efficiently solvable

Prime factoring

$$8249 = ?$$

Computational steps required grow exponentially in the number digits

Hard to compute

Motivation

What is complex and what is efficient?

Multiplication

$$113 \cdot 73 = 8249$$

Number of steps linear in the number of digits

Efficiently solvable

Prime factoring

$$8249 = ?$$

Computational steps required grow exponentially in the number digits

Hard to compute

Can we use the complexity of quantum systems to speedup algorithms?

Guiding questions

Outline of the talk

Which quantum algorithms exist? and *How* do they work?

Deutsch-Jozsa, Grover's Algorithm

Guiding questions

Outline of the talk

Which quantum algorithms exist? and *How* do they work?

Deutsch-Jozsa, Grover’s Algorithm

How much faster are they compared to classical counterpart?

Complexity Theory

Fundamentals of Quantum Computing

Quantum Bit

Classical Bit

$$X \in \{0,1\}$$

Quantum Bit

Classical Bit

$$X \in \{0,1\}$$

$\longrightarrow$

Quantum Bit (Qubit)

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \in \mathscr{H}$$

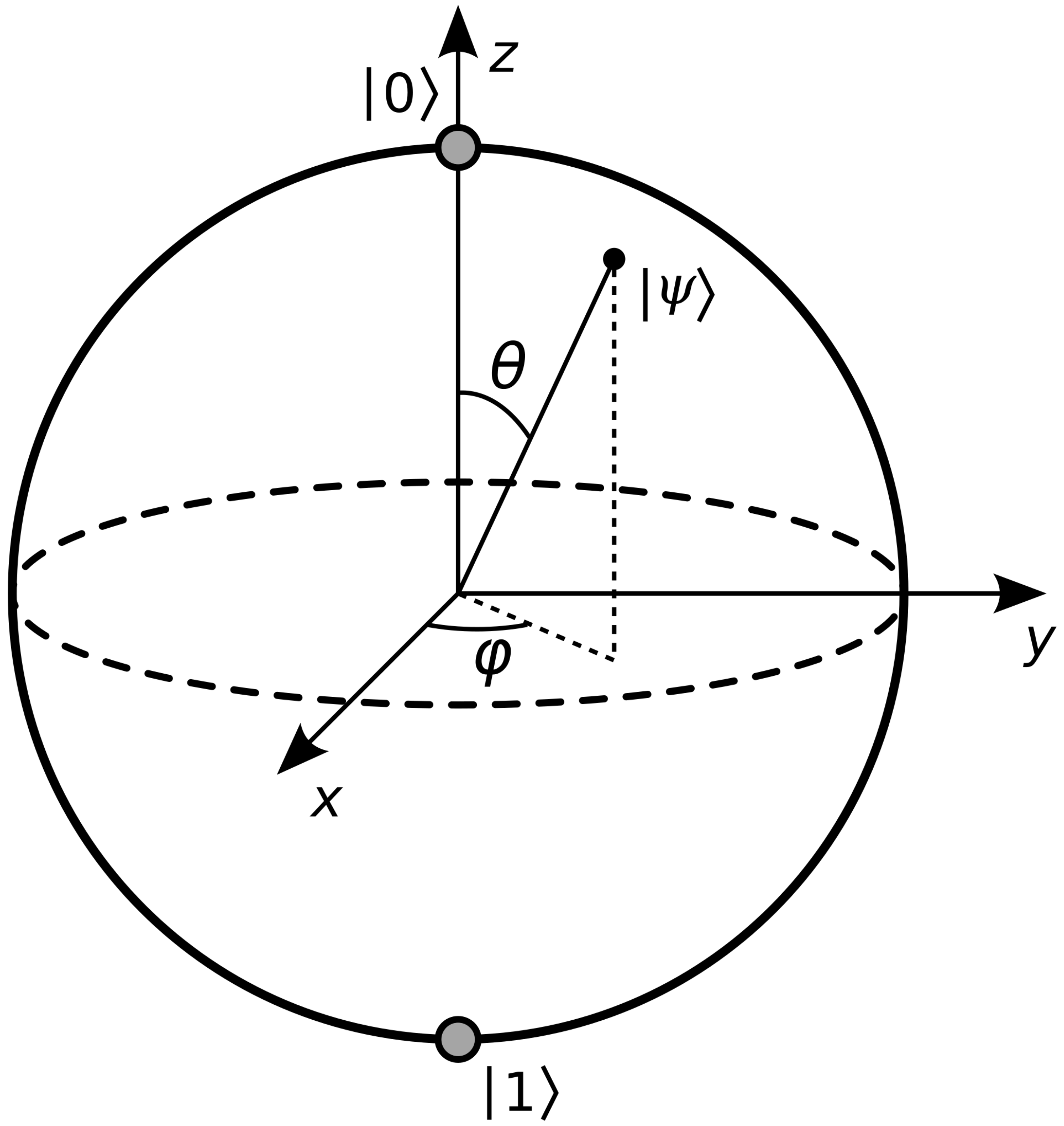
Quantum Bit

Classical Bit

$$X \in \{0,1\}$$

Quantum Bit (Qubit)

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \in \mathscr{H}$$



Quantum Bit

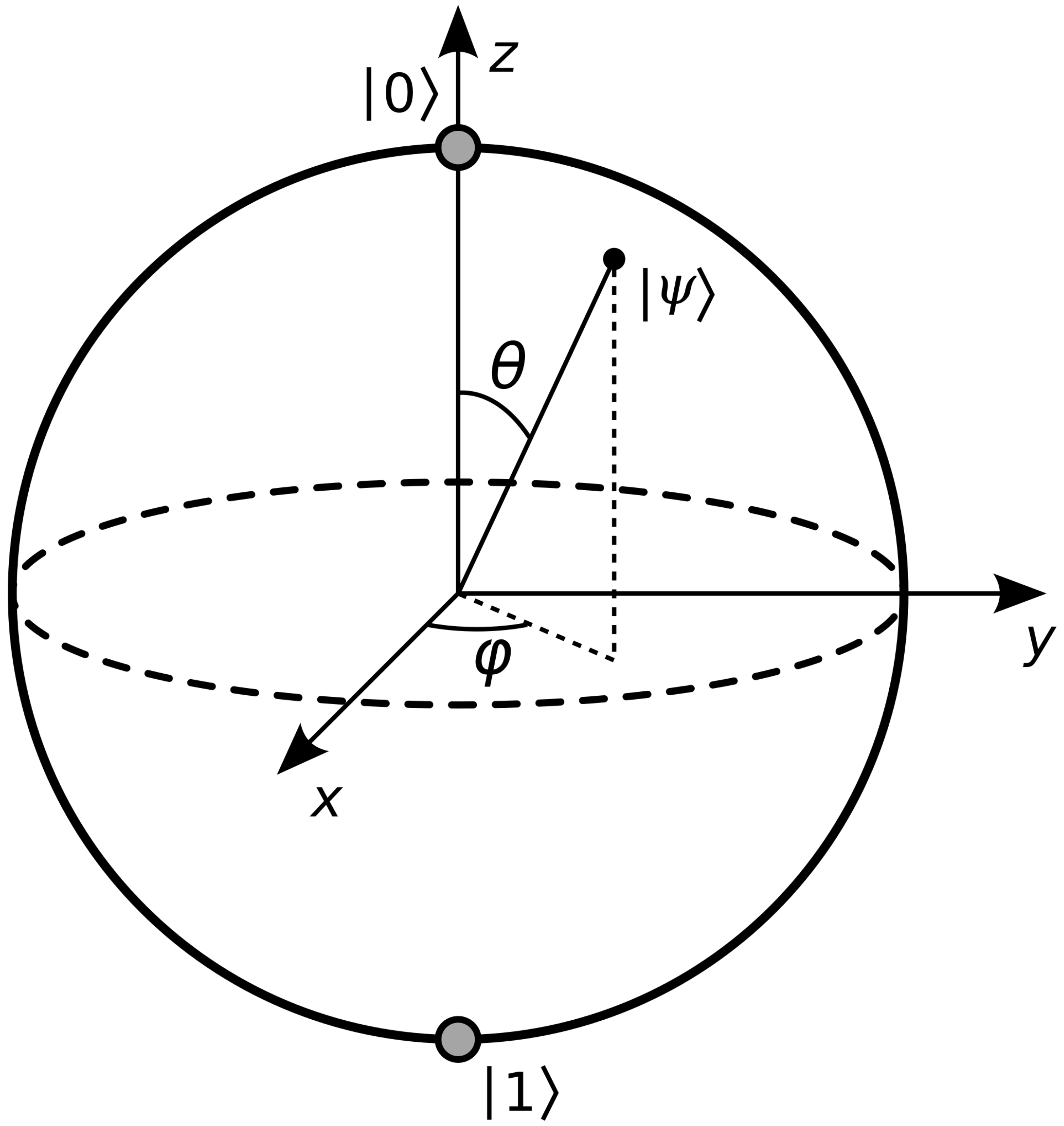
Classical Bit

$$X \in \{0,1\}$$

Quantum Bit (Qubit)

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \in \mathscr{H}$$

In principle Qubit stores *infinite* amount of information



Quantum Bit

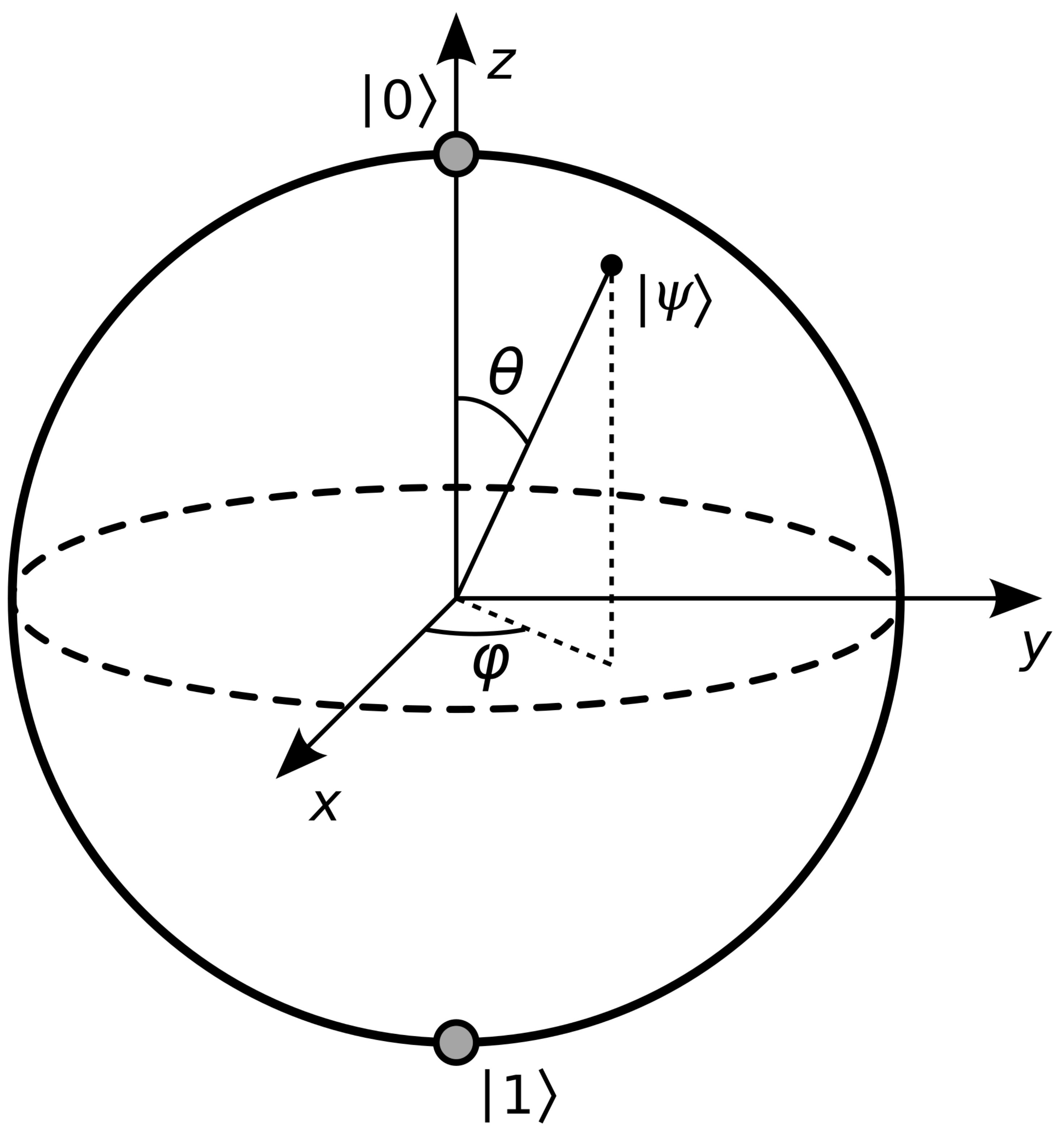
Classical Bit → **Quantum Bit (Qubit)**

$$X \in \{0,1\} \longrightarrow |\psi\rangle = \alpha|0\rangle + \beta|1\rangle \in \mathscr{H}$$

In principle Qubit stores *infinite* amount of information

→ measurement yields a classical Bit

Upon measurement only a *finite* amount is attainable



Quantum Bit

Classical Bit

$$X \in \{0,1\}$$

$\longrightarrow$

Quantum Bit (Qubit)

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \in \mathscr{H}$$

Classical Register

$$S \in \{0,1\}^n$$

e.g. $S = 1001$

Quantum Bit

Classical Bit

$X \in \{0,1\}$

$\longrightarrow$

Quantum Bit (Qubit)

$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \in \mathscr{H}$

Classical Register

$S \in \{0,1\}^n$

e.g. $S = 1001$

$\longrightarrow$

Quantum Register

$|\psi\rangle = |\psi_1\rangle \otimes \ldots \otimes |\psi_n\rangle \in \mathscr{H}^{\otimes n}$

Quantum Bit

Classical Bit

$$X \in \{0,1\}$$

$\longrightarrow$

Quantum Bit (Qubit)

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \in \mathscr{H}$$

Classical Register

$$S \in \{0,1\}^n$$

e.g. $S = 1001$

$\longrightarrow$

Quantum Register

$$|\psi\rangle = |\psi_1\rangle \otimes \ldots \otimes |\psi_n\rangle \in \mathscr{H}^{\otimes n}$$

We will use the short hand notation

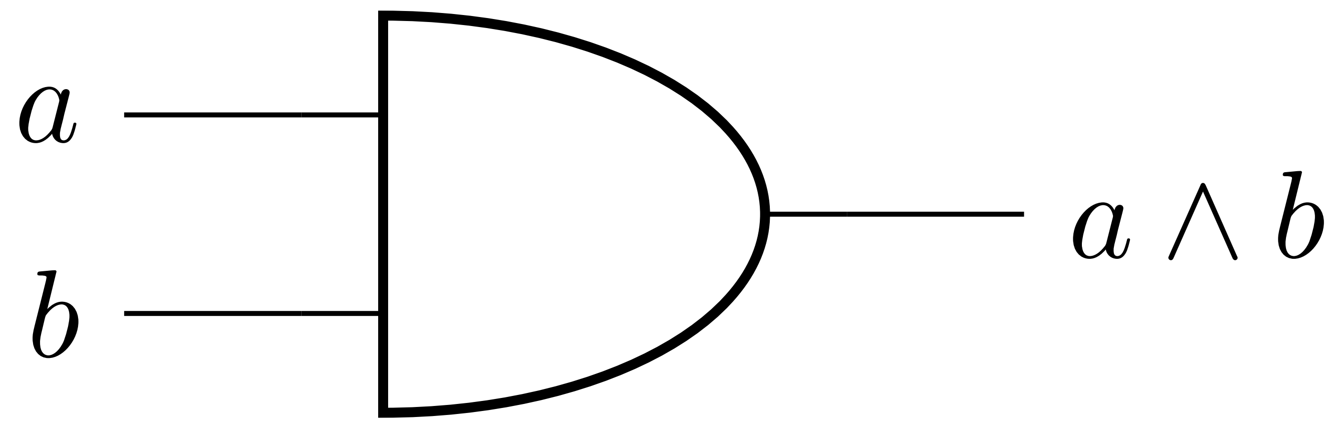
$$|0\rangle \otimes |1\rangle \otimes |0\rangle = |010\rangle$$

Classical logic gates

Processing of classical bits can be described in the language of *gates*

Classical logic gates

Processing of classical bits can be described in the language of *gates*

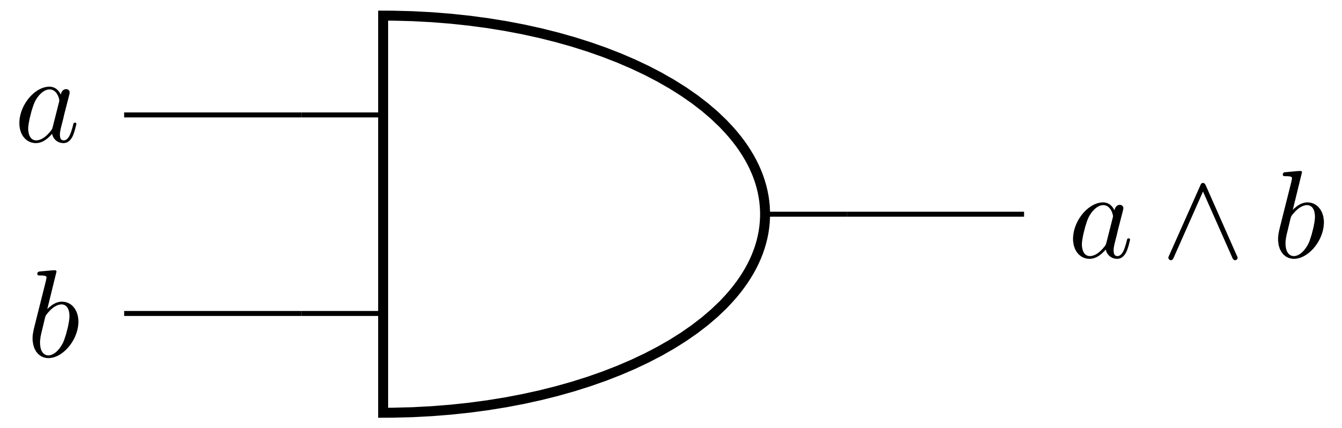


AND Gate

a	b	$a \wedge b$
0	0	0
0	1	0
1	0	0
1	1	1

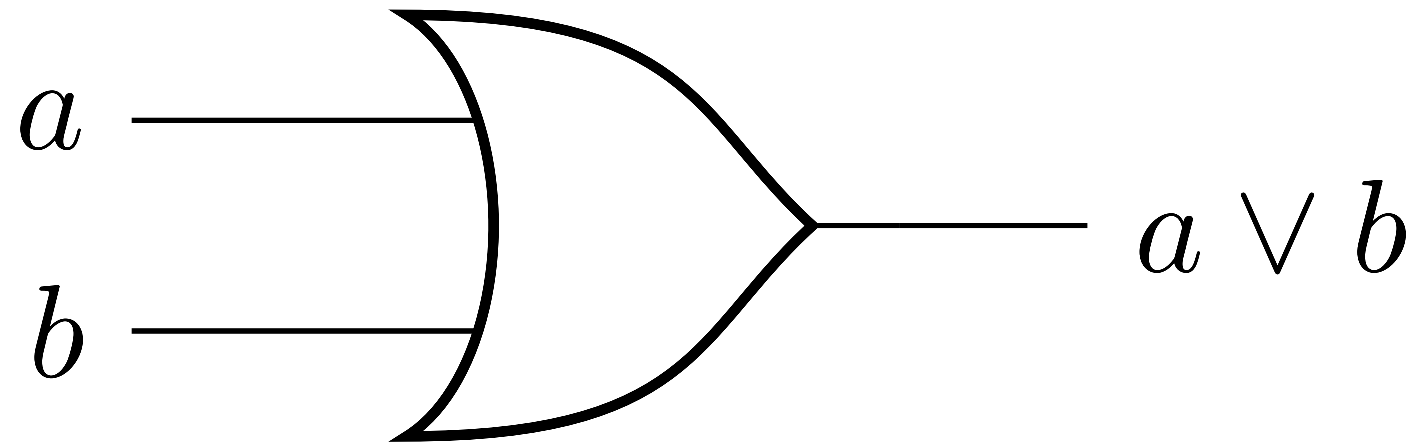
Classical logic gates

Processing of classical bits can be described in the language of *gates*



AND Gate

a	b	$a \wedge b$
0	0	0
0	1	0
1	0	0
1	1	1

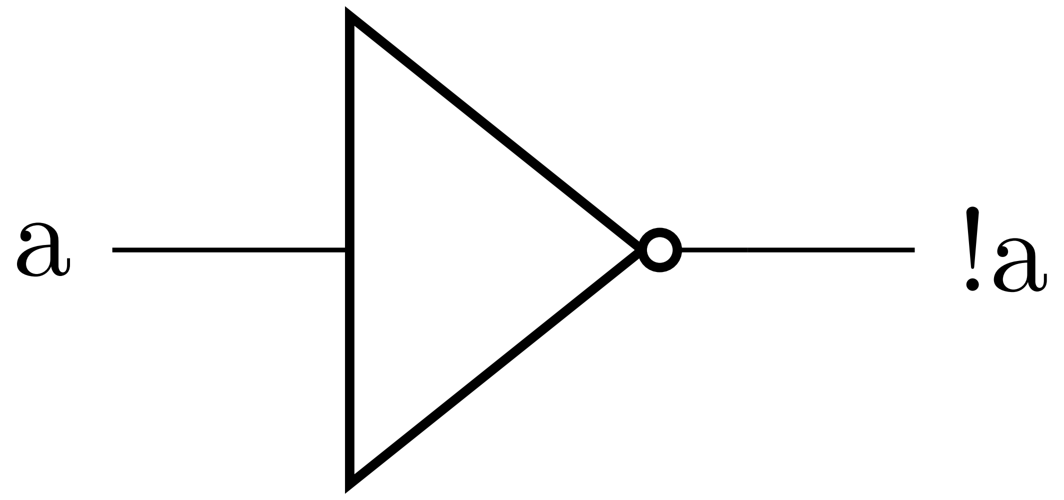


OR Gate

a	b	$a \vee b$
0	0	0
0	1	1
1	0	1
1	1	1

Classical logic gates

Processing of classical bits can be described in the language of *gates*

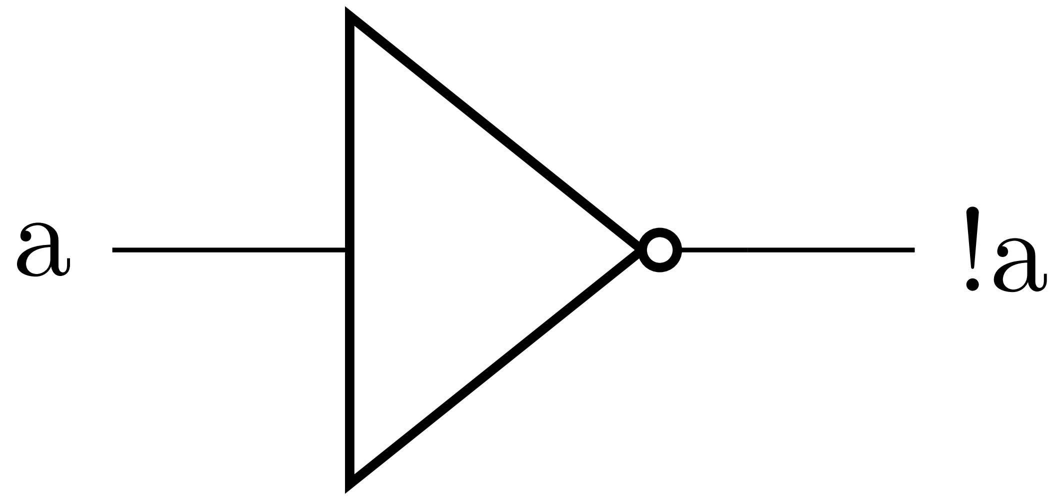


NOT Gate

a	$!a$
0	1
1	0

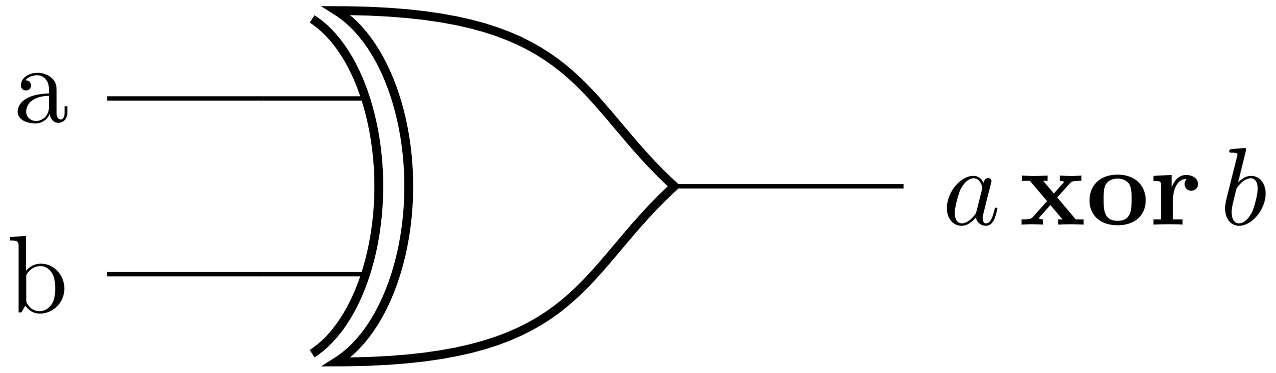
Classical logic gates

Processing of classical bits can be described in the language of *gates*



NOT Gate

a	$!a$
0	1
1	0



XOR Gate

a	b	a **xor** b
0	0	0
0	1	1
1	0	1
1	1	0

Quantum Gates

In non-relativistic QM we can either

Quantum Gates

In non-relativistic QM we can either

evolve in time

$$i\frac{\partial}{\partial t}|\psi\rangle = \hat{H}|\psi\rangle$$

Unitary operations

Quantum Gates

In non-relativistic QM we can either

evolve in time

$$i\frac{\partial}{\partial t}|\psi\rangle = \hat{H}|\psi\rangle$$

Unitary operations

or

measure

$$\hat{O}|\psi\rangle = o|\psi\rangle$$

Hermitian operations

Quantum Gates

In non-relativistic QM we can either

evolve in time

$$i\frac{\partial}{\partial t}|\psi\rangle = \hat{H}|\psi\rangle$$

Unitary operations

or

measure

$$\hat{O}|\psi\rangle = o|\psi\rangle$$

Hermitian operations

Since the theory is <u>linear</u> all quantum gates need to be linear

Quantum Gates

Hadamard Gate

$$\alpha|0\rangle + \beta|1\rangle \quad -\boxed{H}- \quad \frac{\alpha}{\sqrt{2}}(|0\rangle + |1\rangle) + \frac{\beta}{\sqrt{2}}(|0\rangle - |1\rangle)$$

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

Quantum Gates

Hadamard Gate

$$\alpha|0\rangle + \beta|1\rangle \quad -\boxed{H}- \quad \frac{\alpha}{\sqrt{2}}(|0\rangle + |1\rangle) + \frac{\beta}{\sqrt{2}}(|0\rangle - |1\rangle)$$

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

X, Y, Z Gates

$$\alpha|0\rangle + \beta|1\rangle \quad -\boxed{X}- \quad \alpha|1\rangle + \beta|0\rangle$$

Quantum Gates

Hadamard Gate

$$\alpha|0\rangle + \beta|1\rangle \quad -\!\boxed{H}\!- \quad \frac{\alpha}{\sqrt{2}}(|0\rangle + |1\rangle) + \frac{\beta}{\sqrt{2}}(|0\rangle - |1\rangle)$$

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

X, Y, Z Gates

$$\alpha|0\rangle + \beta|1\rangle \quad -\!\boxed{X}\!- \quad \alpha|1\rangle + \beta|0\rangle$$

$$\alpha|0\rangle + \beta|1\rangle \quad -\!\boxed{Y}\!- \quad \alpha i|1\rangle - \beta i|0\rangle$$

Quantum Gates

Hadamard Gate

$$\alpha|0\rangle + \beta|1\rangle \;\xrightarrow{\;\boxed{H}\;}\; \frac{\alpha}{\sqrt{2}}(|0\rangle + |1\rangle) + \frac{\beta}{\sqrt{2}}(|0\rangle - |1\rangle)$$

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

X, Y, Z Gates

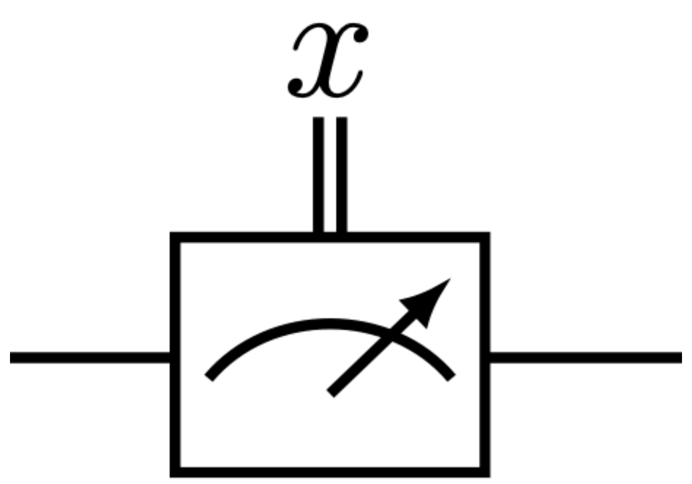
$$\alpha|0\rangle + \beta|1\rangle \;\xrightarrow{\;\boxed{X}\;}\; \alpha|1\rangle + \beta|0\rangle$$

$$\alpha|0\rangle + \beta|1\rangle \;\xrightarrow{\;\boxed{Y}\;}\; \alpha i|1\rangle - \beta i|0\rangle$$

$$\alpha|0\rangle + \beta|1\rangle \;\xrightarrow{\;\boxed{Z}\;}\; \alpha|0\rangle - \beta|1\rangle$$

Quantum Gates

Measurement Gate

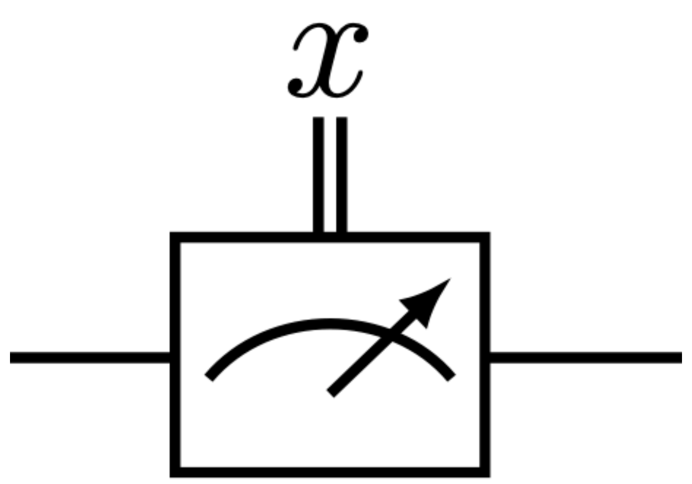


$$x \in \{-1,1\}$$

$$\hat{Z}|\psi\rangle = x|\psi\rangle$$

Quantum Gates

Measurement Gate

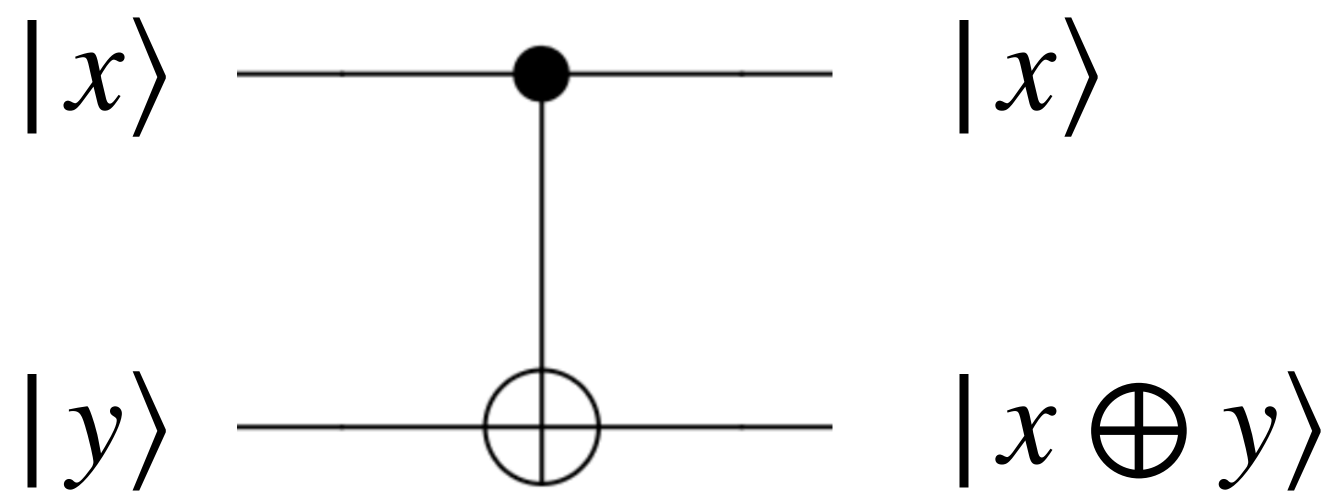


$x \in \{-1, 1\}$

$\hat{Z}|\psi\rangle = x|\psi\rangle$

Control Gates

flip the second qubit if the first is $|1\rangle$

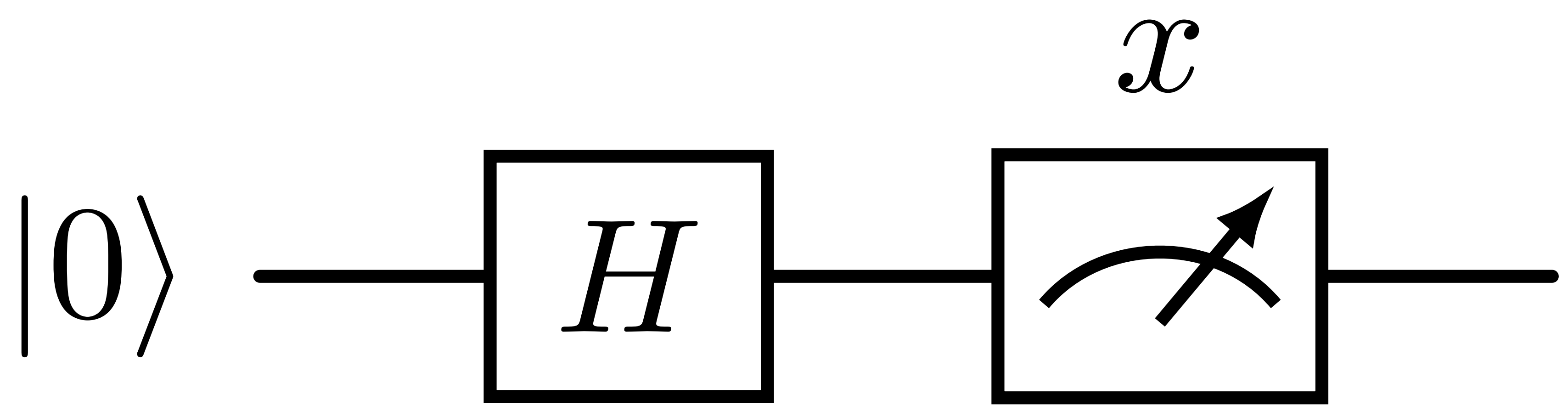


$0 \oplus 0 = 0$
$1 \oplus 0 = 1$
$0 \oplus 1 = 1$
$1 \oplus 1 = 0$

Quantum Circuits

Similar to classical circuits, we can connect Quantum gates

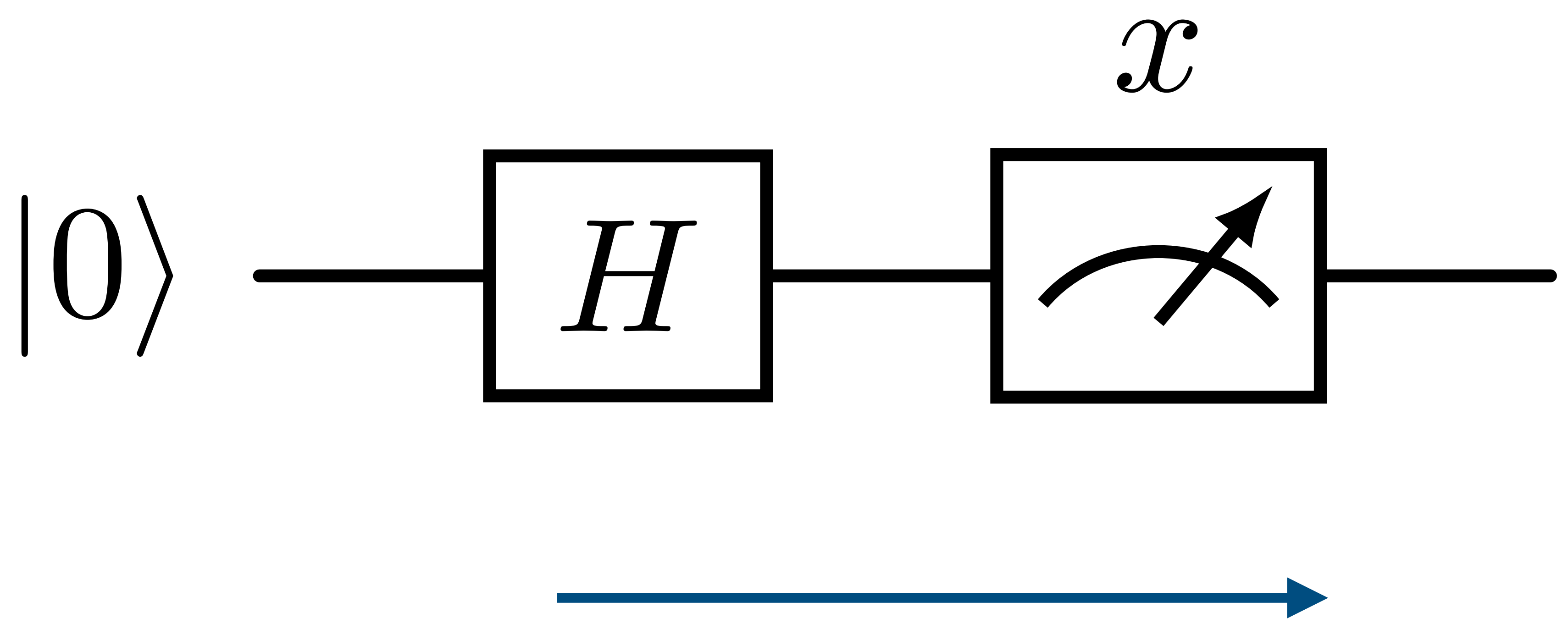
Example: Random number generator



Quantum Circuits

Similar to classical circuits, we can connect Quantum gates

Example: Random number generator

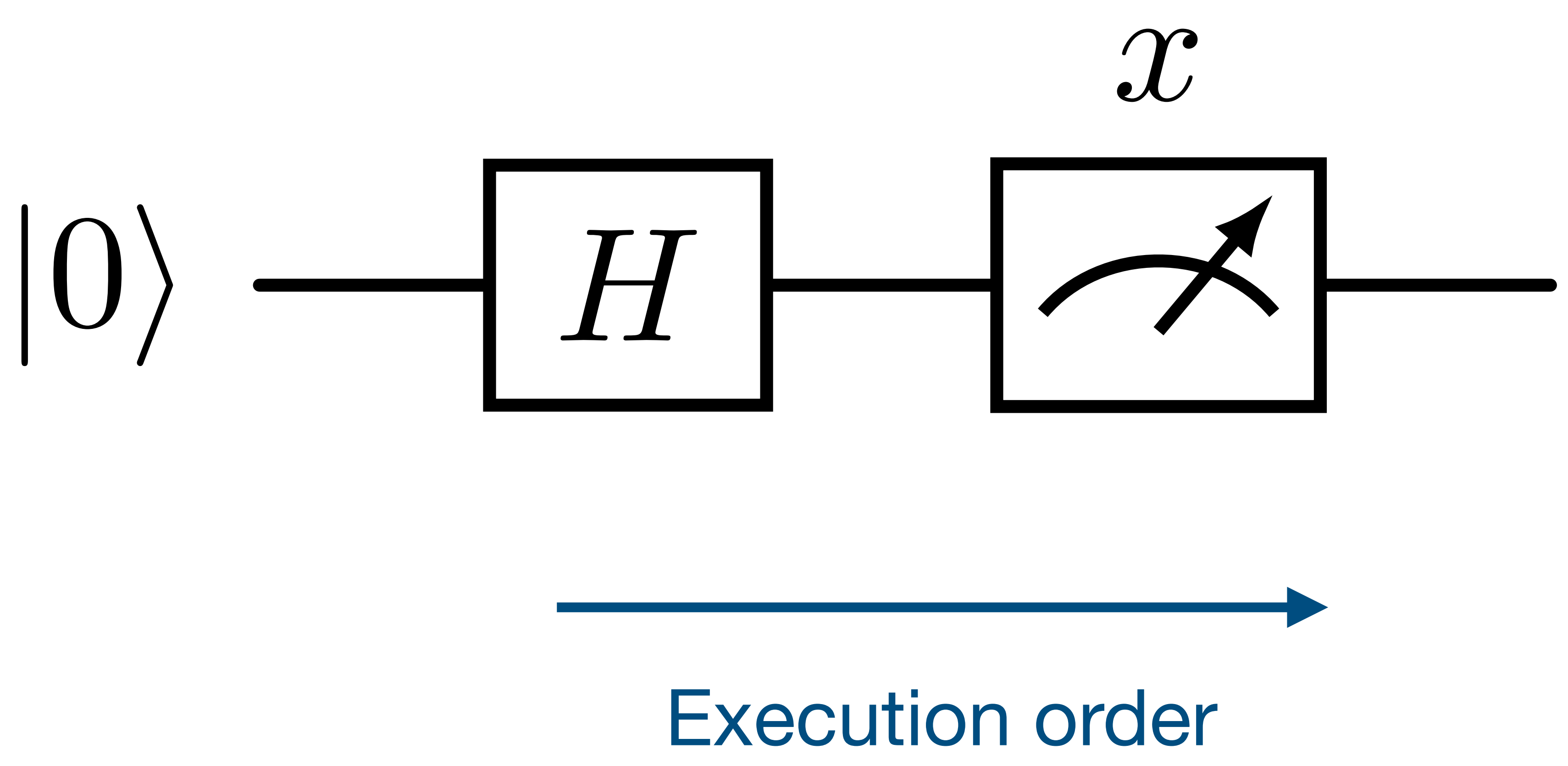


Execution order

Quantum Circuits

Similar to classical circuits, we can connect Quantum gates

Example: Random number generator

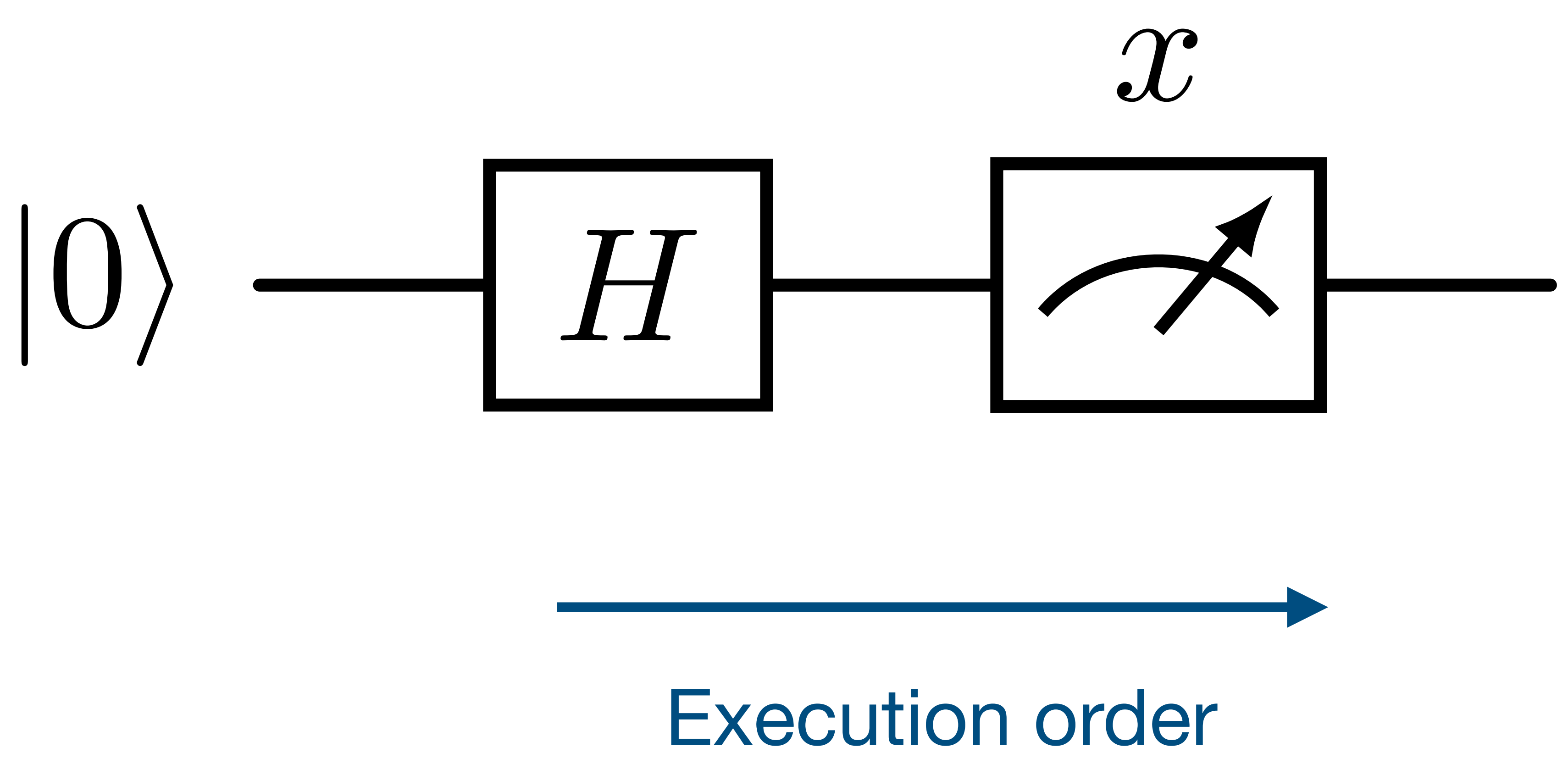


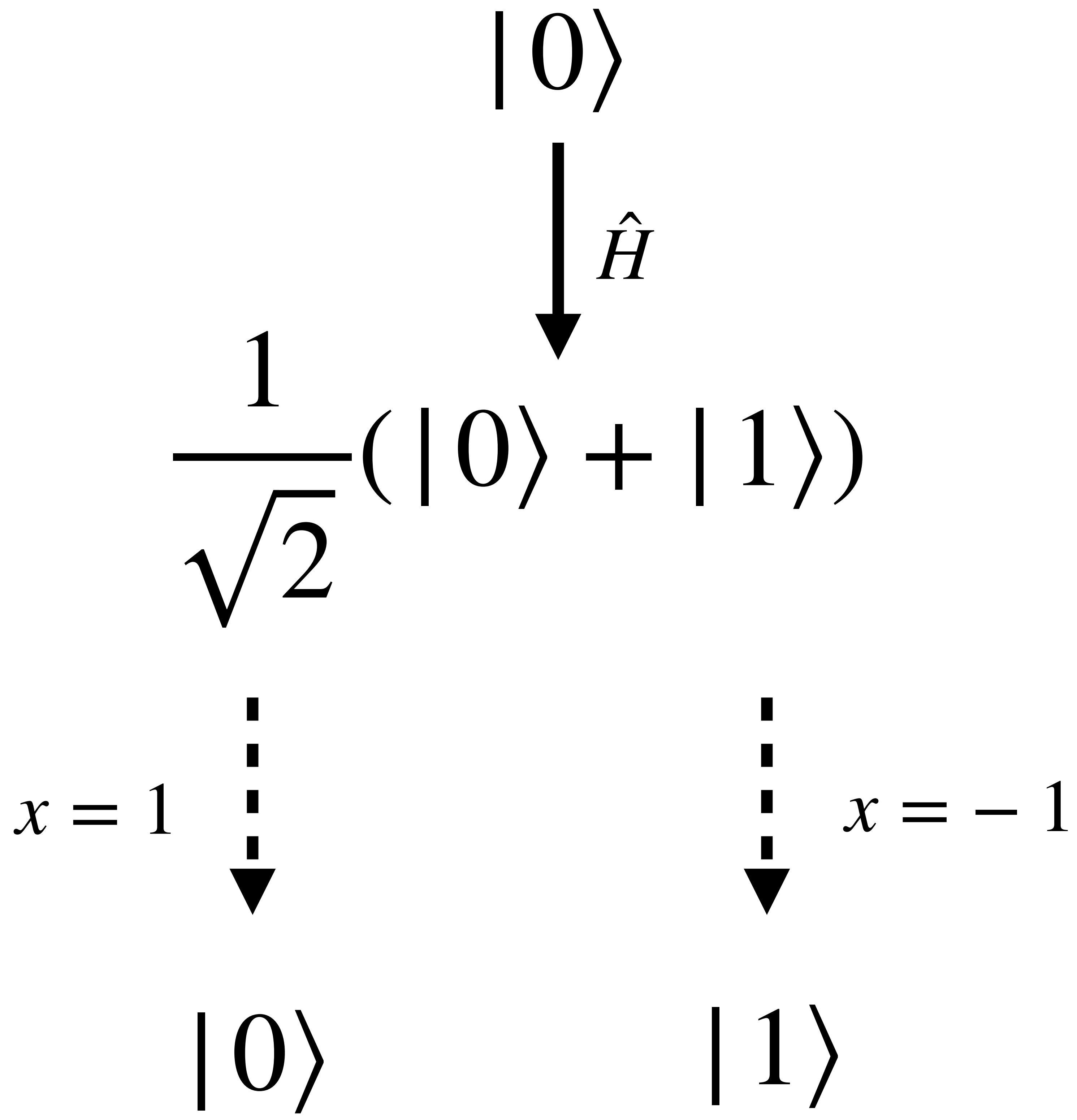
$$|0\rangle \xrightarrow{\hat{H}} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$

Quantum Circuits

Similar to classical circuits, we can connect Quantum gates

Example: Random number generator

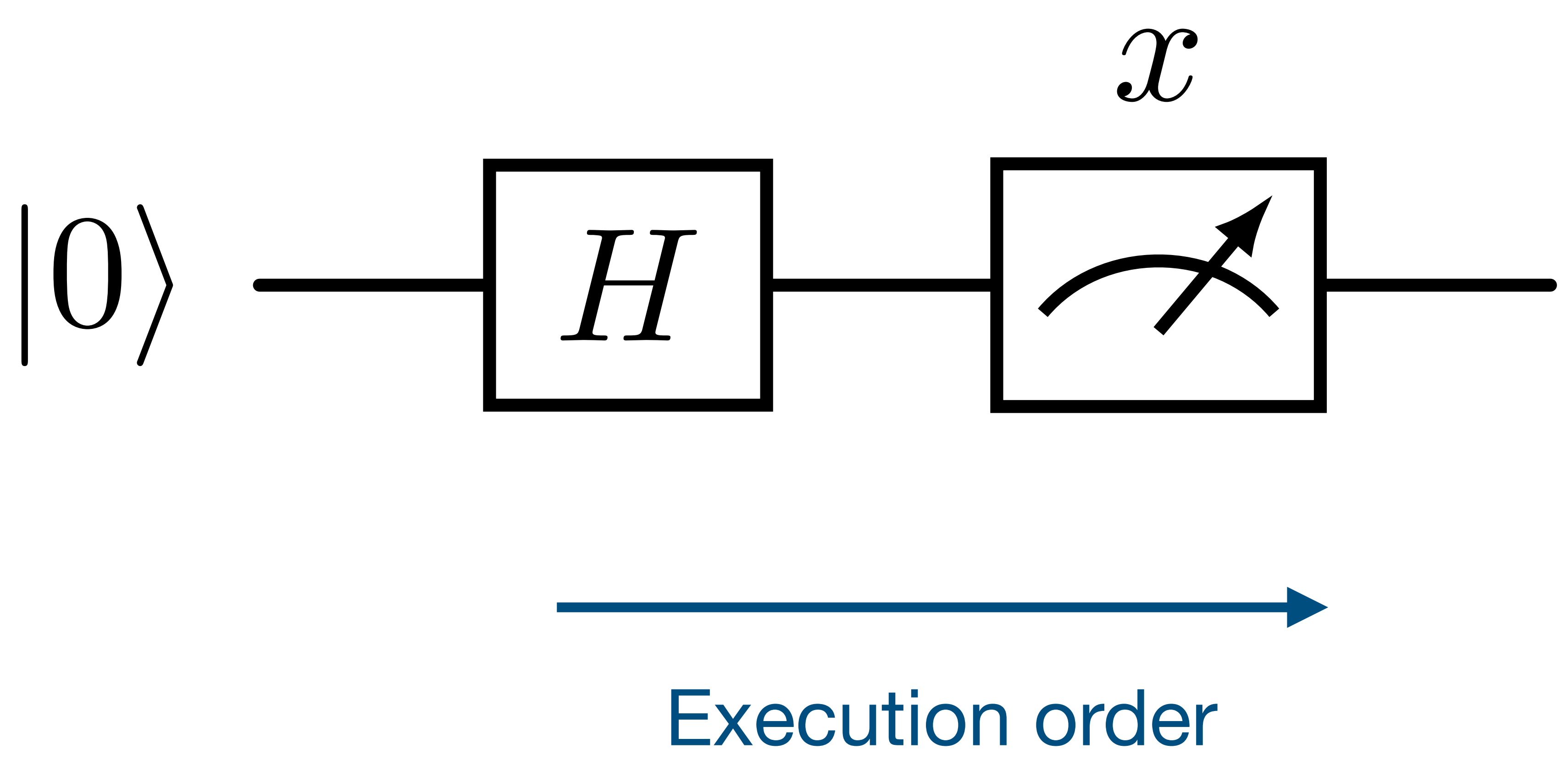


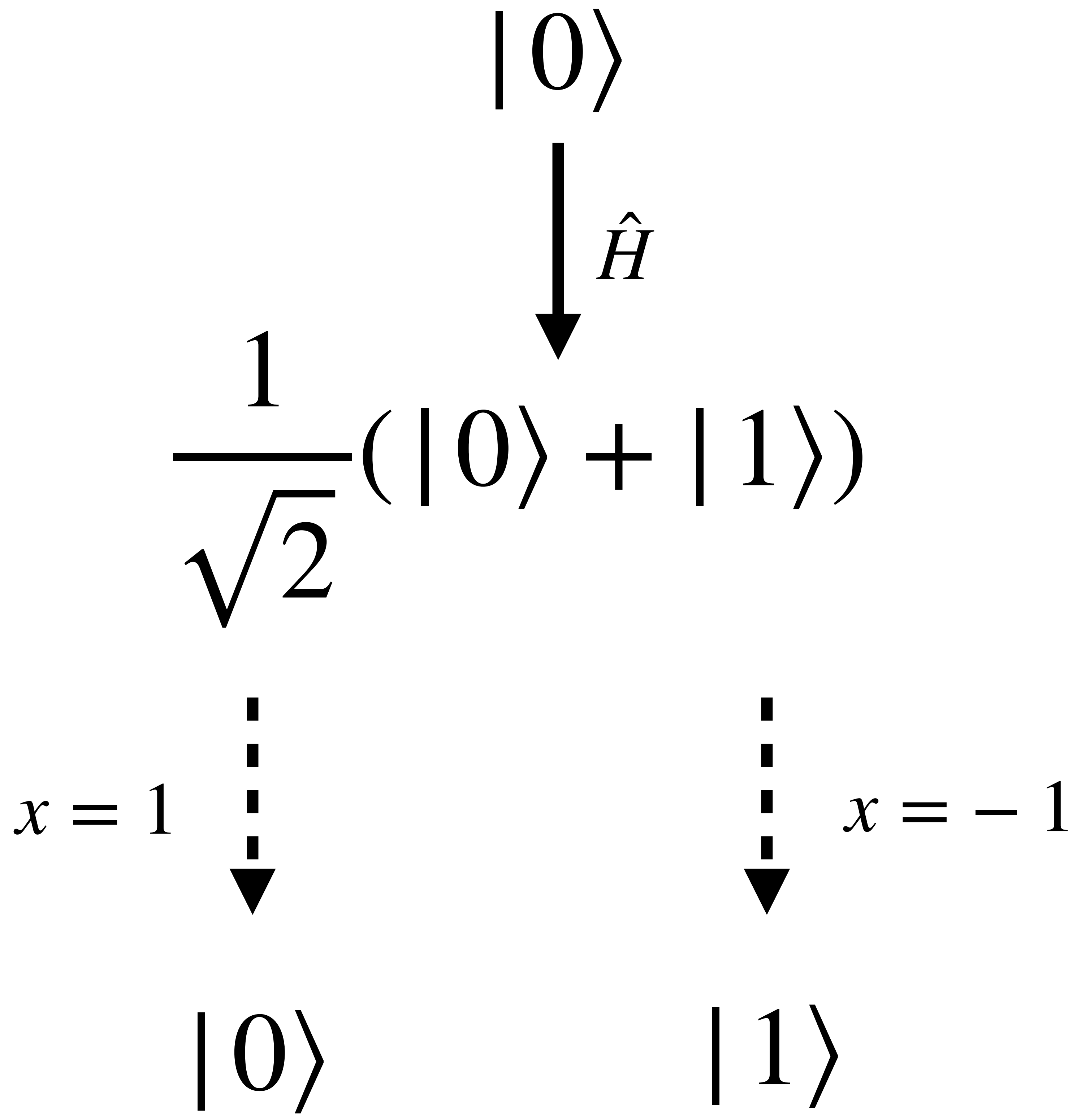


Quantum Circuits

Similar to classical circuits, we can connect Quantum gates

Example: Random number generator





Result of measurement is intrinsically random → sampling necessary

No Cloning Theorem

Suppose $\hat{U} : \mathscr{H} \rightarrow \mathscr{H}$ clones the first qubit into second

$$\hat{U}(|\phi\rangle \otimes |\psi\rangle) = |\phi\rangle \otimes |\phi\rangle$$

No Cloning Theorem

Suppose $\hat{U} : \mathscr{H} \to \mathscr{H}$ clones the first qubit into second

$$\hat{U}(|\phi\rangle \otimes |\psi\rangle) = |\phi\rangle \otimes |\phi\rangle$$

but due to linearity of $\hat{U}$

$$\hat{U}(a|0\rangle + b|1\rangle) \otimes |0\rangle = a\hat{U}|00\rangle + b\hat{U}|10\rangle =$$

No Cloning Theorem

Suppose $\hat{U} : \mathscr{H} \to \mathscr{H}$ clones the first qubit into second

$$\hat{U}(|\phi\rangle \otimes |\psi\rangle) = |\phi\rangle \otimes |\phi\rangle$$

but due to linearity of $\hat{U}$

$$\hat{U}(a|0\rangle + b|1\rangle) \otimes |0\rangle = a\hat{U}|00\rangle + b\hat{U}|10\rangle = a|00\rangle + b|11\rangle$$

No Cloning Theorem

Suppose $\hat{U} : \mathscr{H} \to \mathscr{H}$ clones the first qubit into second

$$\hat{U}(|\phi\rangle \otimes |\psi\rangle) = |\phi\rangle \otimes |\phi\rangle$$

but due to linearity of $\hat{U}$

$$\hat{U}(a|0\rangle + b|1\rangle) \otimes |0\rangle = a\hat{U}|00\rangle + b\hat{U}|10\rangle = a|00\rangle + b|11\rangle$$

in contradiction to

$$\hat{U}(a|0\rangle + b|1\rangle) \otimes |0\rangle = (a|0\rangle + b|1\rangle) \otimes (a|0\rangle + b|1\rangle)$$

No Cloning Theorem

Suppose $\hat{U} : \mathscr{H} \to \mathscr{H}$ clones the first qubit into second

$$\hat{U}(|\phi\rangle \otimes |\psi\rangle) = |\phi\rangle \otimes |\phi\rangle$$

but due to linearity of $\hat{U}$

$$\hat{U}(a|0\rangle + b|1\rangle) \otimes |0\rangle = a\hat{U}|00\rangle + b\hat{U}|10\rangle = a|00\rangle + b|11\rangle$$

in contradiction to

$$\hat{U}(a|0\rangle + b|1\rangle) \otimes |0\rangle = (a|0\rangle + b|1\rangle) \otimes (a|0\rangle + b|1\rangle)$$

Therefore: Quantum circuits cannot split a qubit into two

Deutsch-Jozsa Algorithm

A coin is considered *real* if it has distinct sides …

Deutsch-Jozsa Algorithm

A coin is considered *real* if it has distinct sides …

… and otherwise *fake*

Deutsch-Jozsa Algorithm

fake

real

Classical: we need check both sides → <u>two</u> measurements

Deutsch-Jozsa Algorithm

fake

real

Classical: we need check both sides → <u>two</u> measurements

Quantum: <u>one</u> measurement sufficient (Deutsch-Jozsa Algorithm)

Deutsch-Jozsa Algorithm

Let $f : \{0,1\} \to \{0,1\}$ be a binary function. We want to know whether

real coin

f is *balanced* ($f(0) \neq f(1)$)

fake coin

f is *constant* ($f(0) = f(1)$)

Deutsch-Jozsa Algorithm

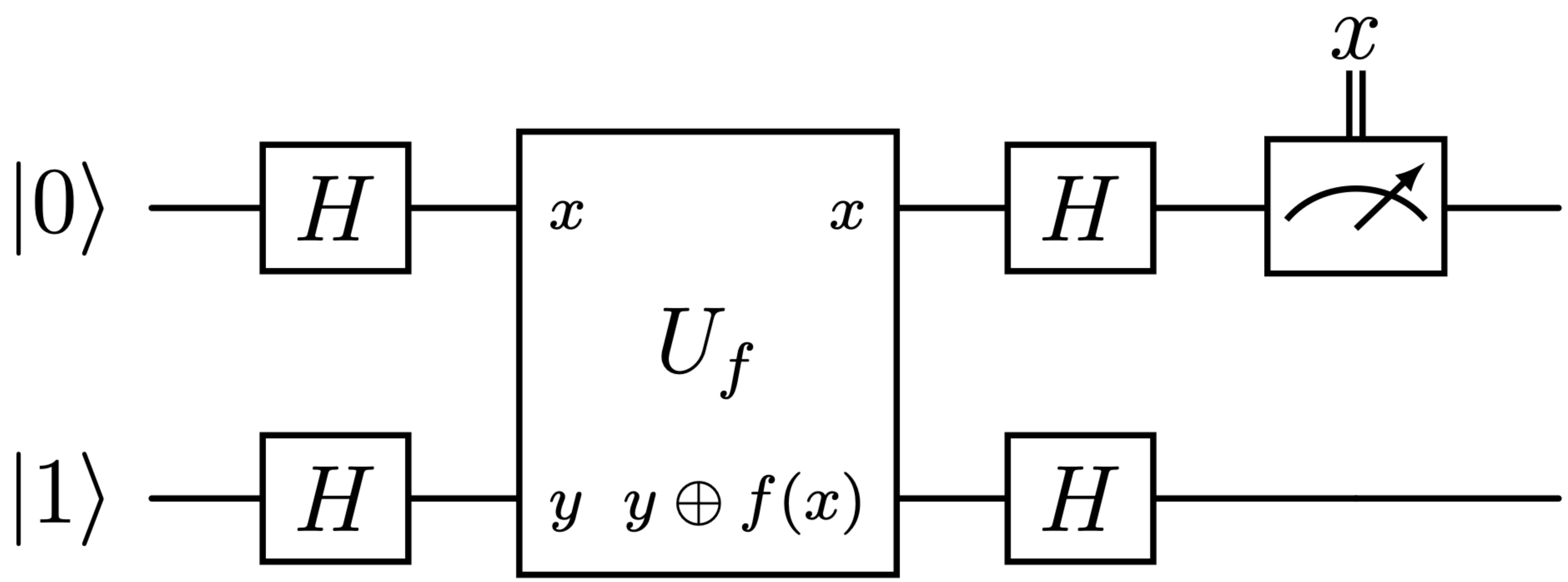
Let $f : \{0,1\} \rightarrow \{0,1\}$ be a binary function. We want to know whether

real coin

f is *balanced* ($f(0) \neq f(1)$)

fake coin

f is *constant* ($f(0) = f(1)$)



Deutsch-Jozsa Algorithm

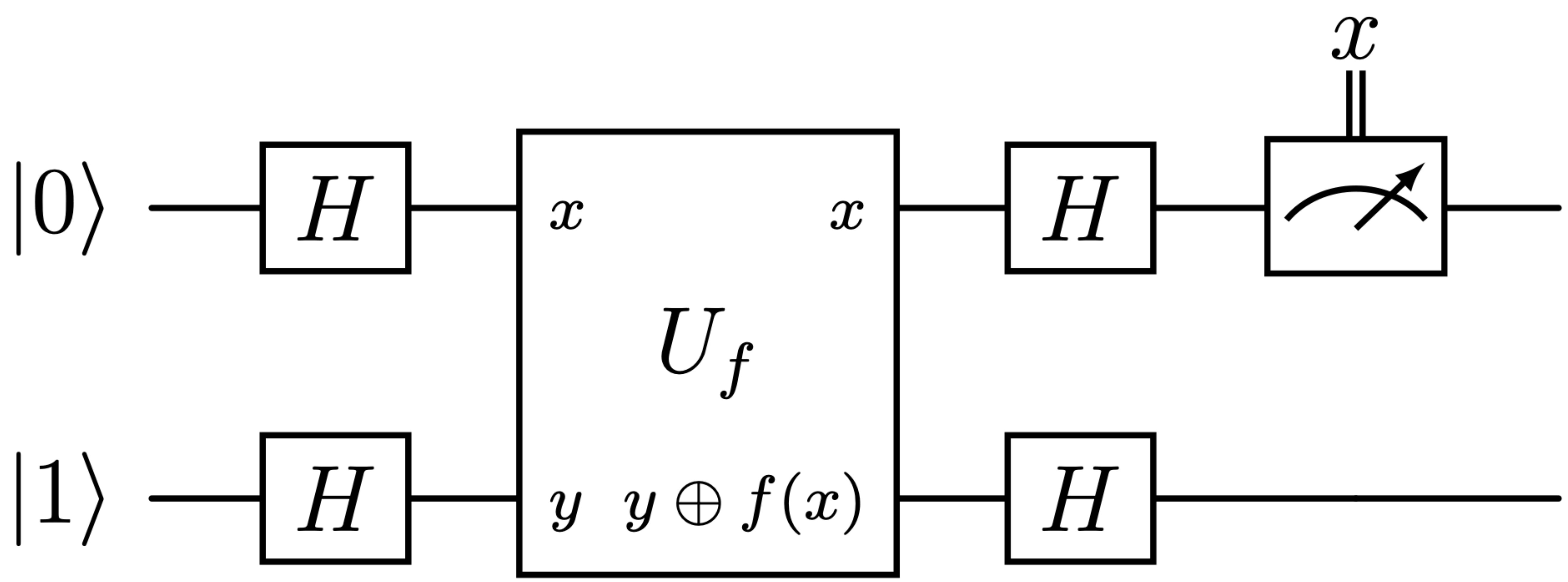
Let $f: \{0,1\} \rightarrow \{0,1\}$ be a binary function. We want to know whether

real coin

f is *balanced* ($f(0) \neq f(1)$)

fake coin

f is *constant* ($f(0) = f(1)$)



$\underline{x = -1}$: f is balanced
real coin

$\underline{x = 1}$: f is constant
fake coin

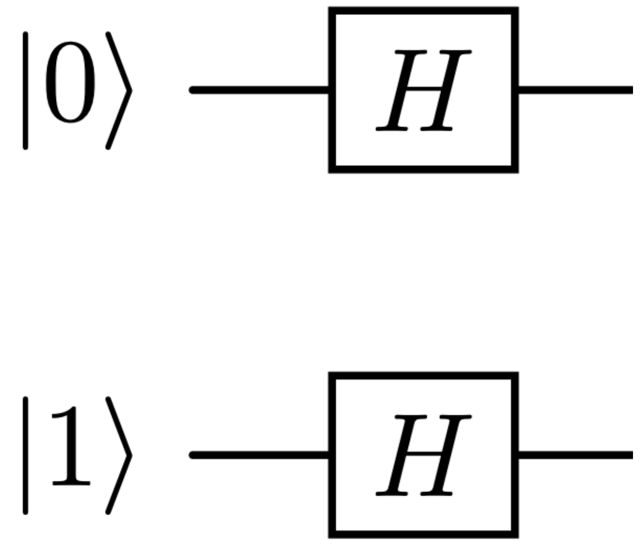
Deutsch-Jozsa Algorithm

$|0\rangle$ —

$|1\rangle$ —

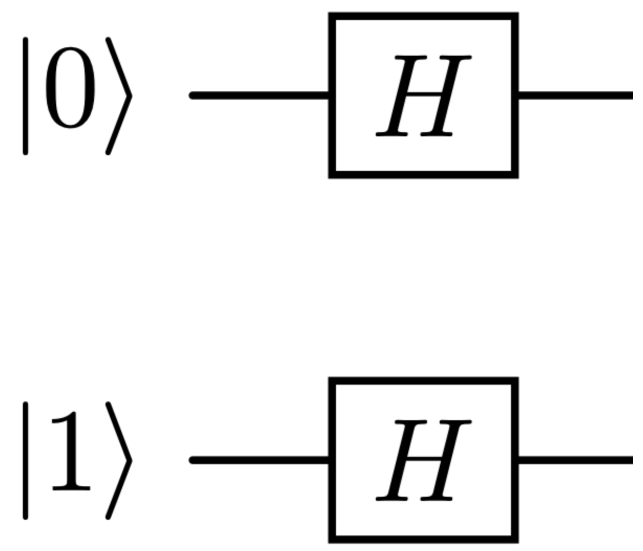
$$|\psi\rangle = |0\rangle \otimes |1\rangle$$

Deutsch-Jozsa Algorithm



$$|\psi\rangle = |0\rangle \otimes |1\rangle \xrightarrow{\hat{H} \otimes \hat{H}} \hat{H}|0\rangle \otimes \hat{H}|1\rangle = \frac{1}{2}\big(|0\rangle + |1\rangle\big)\big(|0\rangle - |1\rangle\big)$$

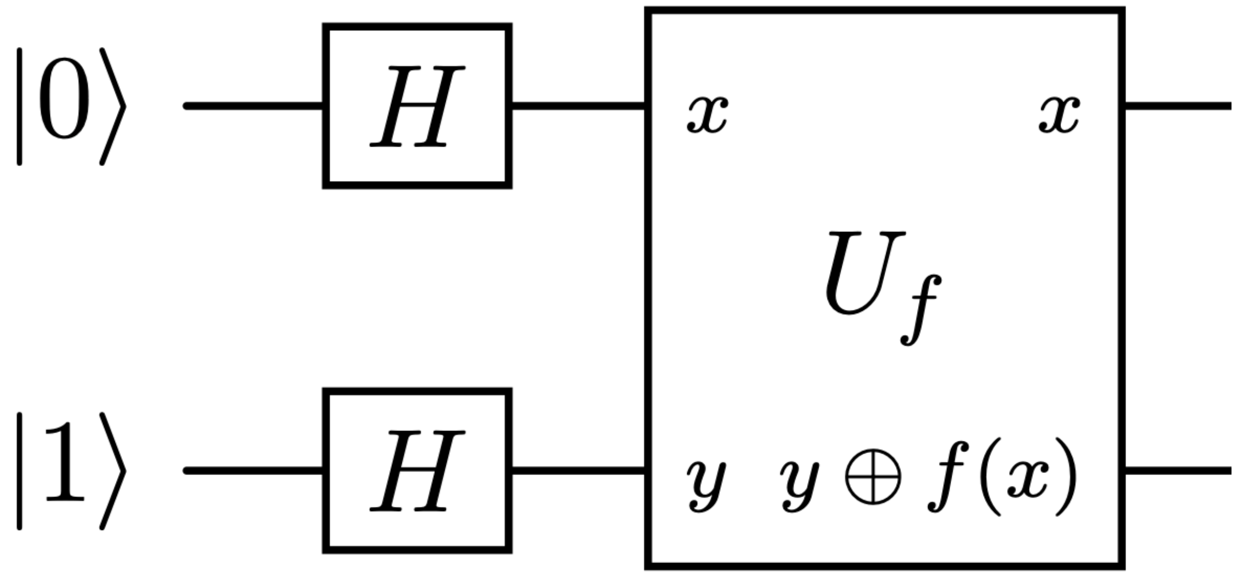
Deutsch-Jozsa Algorithm



$$|\psi\rangle = |0\rangle \otimes |1\rangle \xrightarrow{\hat{H}\otimes\hat{H}} \hat{H}|0\rangle \otimes \hat{H}|1\rangle = \frac{1}{2}\Big(|0\rangle + |1\rangle\Big)\Big(|0\rangle - |1\rangle\Big)$$

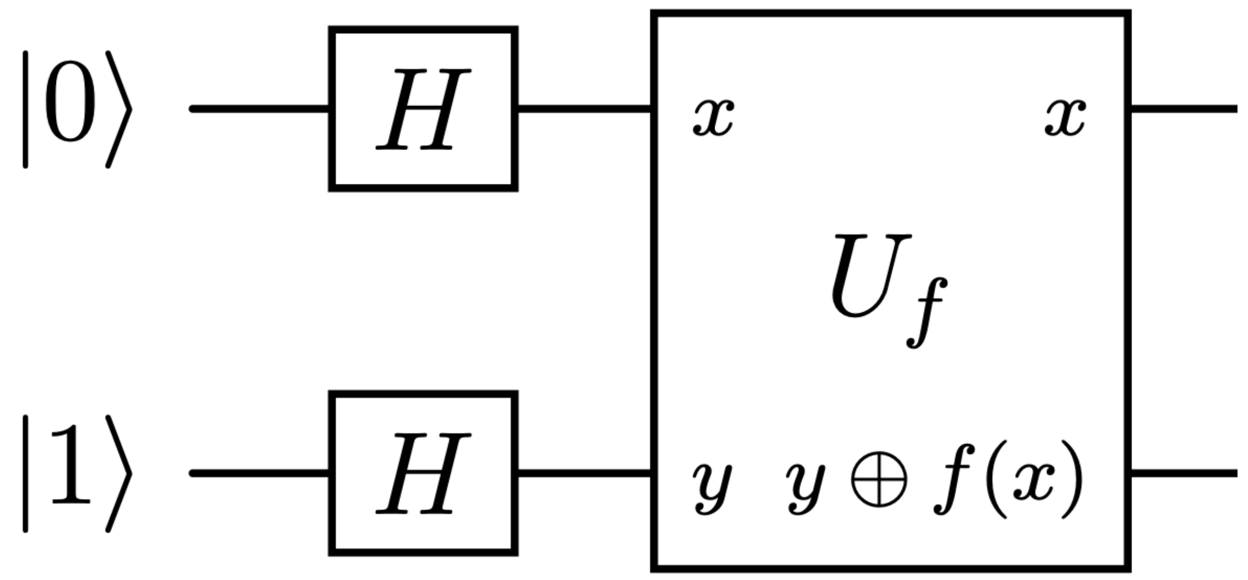
$$= \frac{1}{2}(|00\rangle + |10\rangle - |11\rangle - |01\rangle)$$

Deutsch-Jozsa Algorithm



$$\frac{1}{2}(\,|00\rangle + |10\rangle - |11\rangle - |01\rangle) \xrightarrow{U_f} \frac{1}{2}U_f\big(\,|00\rangle + |10\rangle - |11\rangle - |01\rangle\big)$$

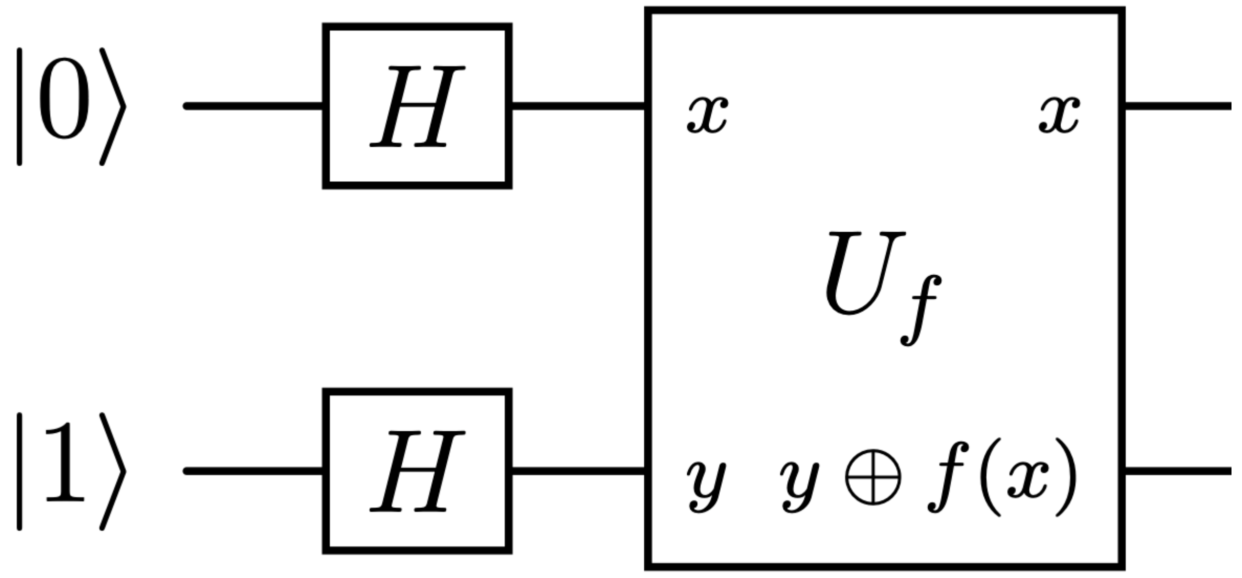
Deutsch-Jozsa Algorithm



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

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

Deutsch-Jozsa Algorithm

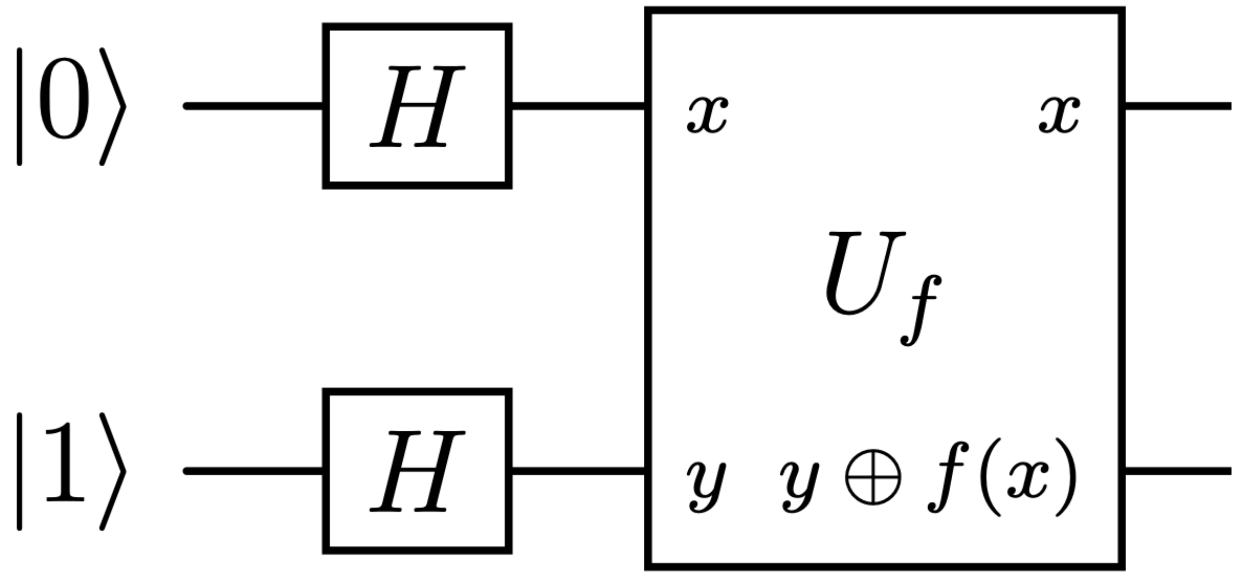


$$U_f|xy\rangle = |x(y \oplus f(x))\rangle$$

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

$$= \frac{1}{2}\big(|0(0 \oplus f(0))\rangle + |1(0 \oplus f(1))\rangle - |1(1 \oplus f(1))\rangle - |0(1 \oplus f(0))\rangle\big)$$

Deutsch-Jozsa Algorithm

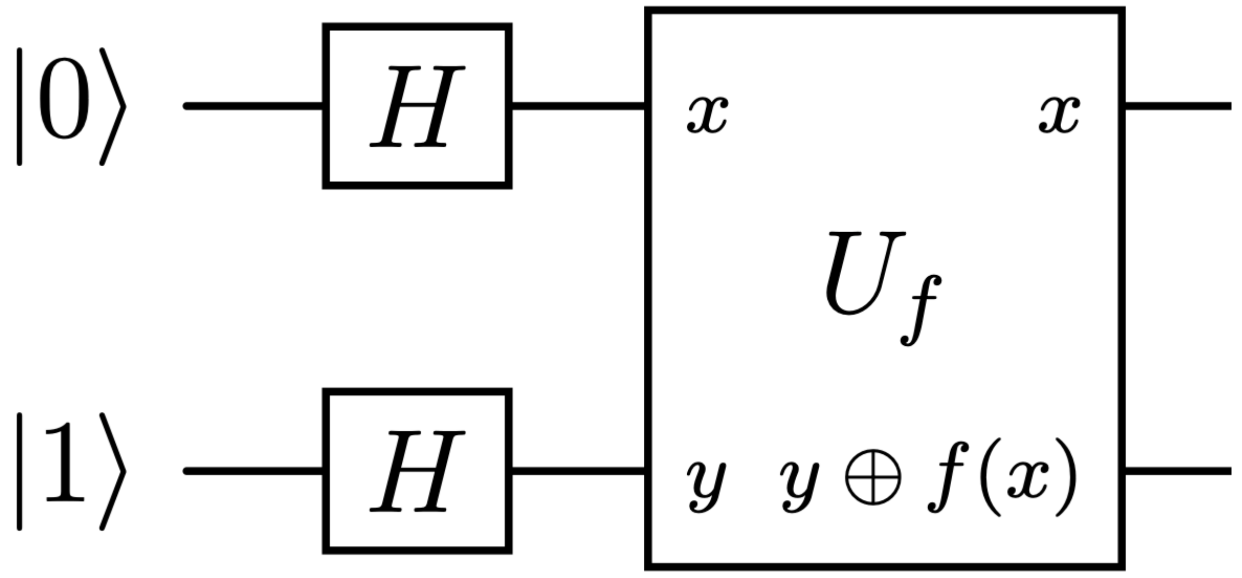


$$U_f|xy\rangle = |x(y \oplus f(x))\rangle$$

$$\frac{1}{2}\left(|0(0 \oplus f(0))\rangle + |1(0 \oplus f(1))\rangle - |1(1 \oplus f(1))\rangle - |0(1 \oplus f(0))\rangle\right)$$

$$= \frac{1}{2}\left((-1)^{f(0)}|0\rangle \otimes (|0\rangle - |1\rangle) + (-1)^{f(1)}|1\rangle \otimes (|0\rangle - |1\rangle)\right)$$

Deutsch-Jozsa Algorithm

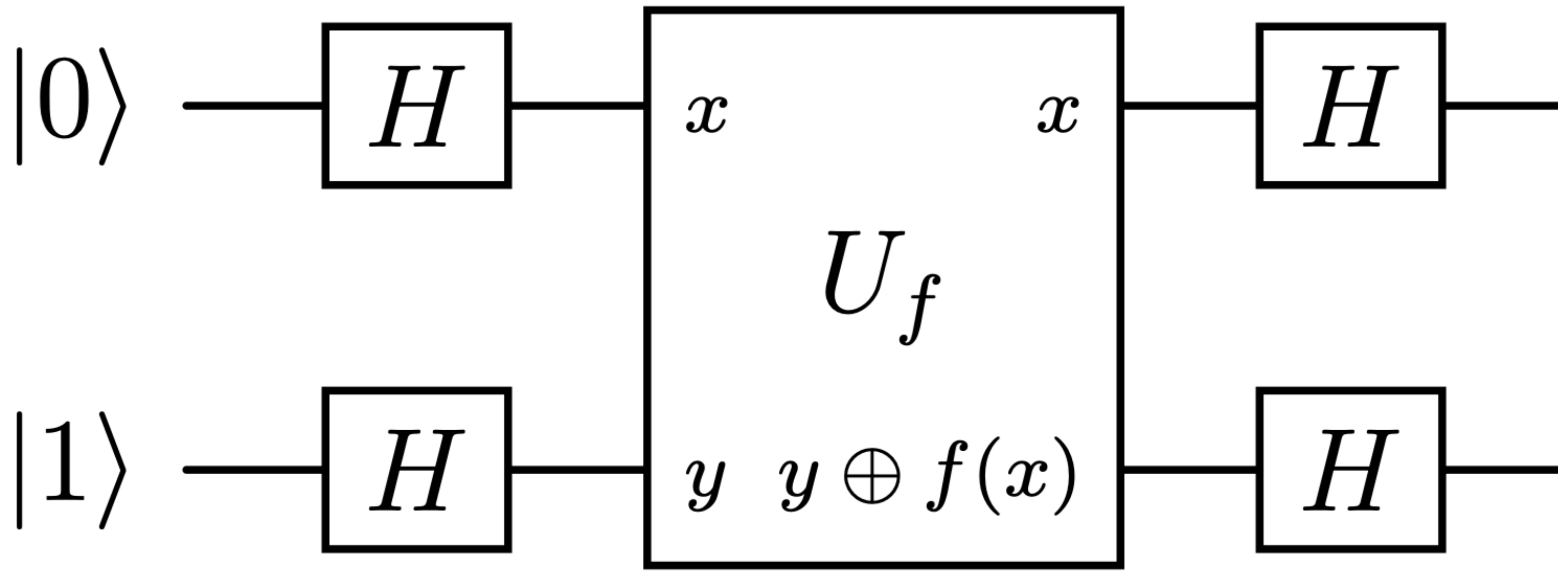


$$U_f|xy\rangle = |x(y \oplus f(x))\rangle$$

$$\frac{1}{2}\Big((-1)^{f(0)}|0\rangle \otimes (|0\rangle - |1\rangle) + (-1)^{f(1)}|1\rangle \otimes (|0\rangle - |1\rangle)\Big)$$

$$= \frac{1}{2}\Big(((-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle) \otimes (|0\rangle - |1\rangle)\Big)$$

Deutsch-Jozsa Algorithm



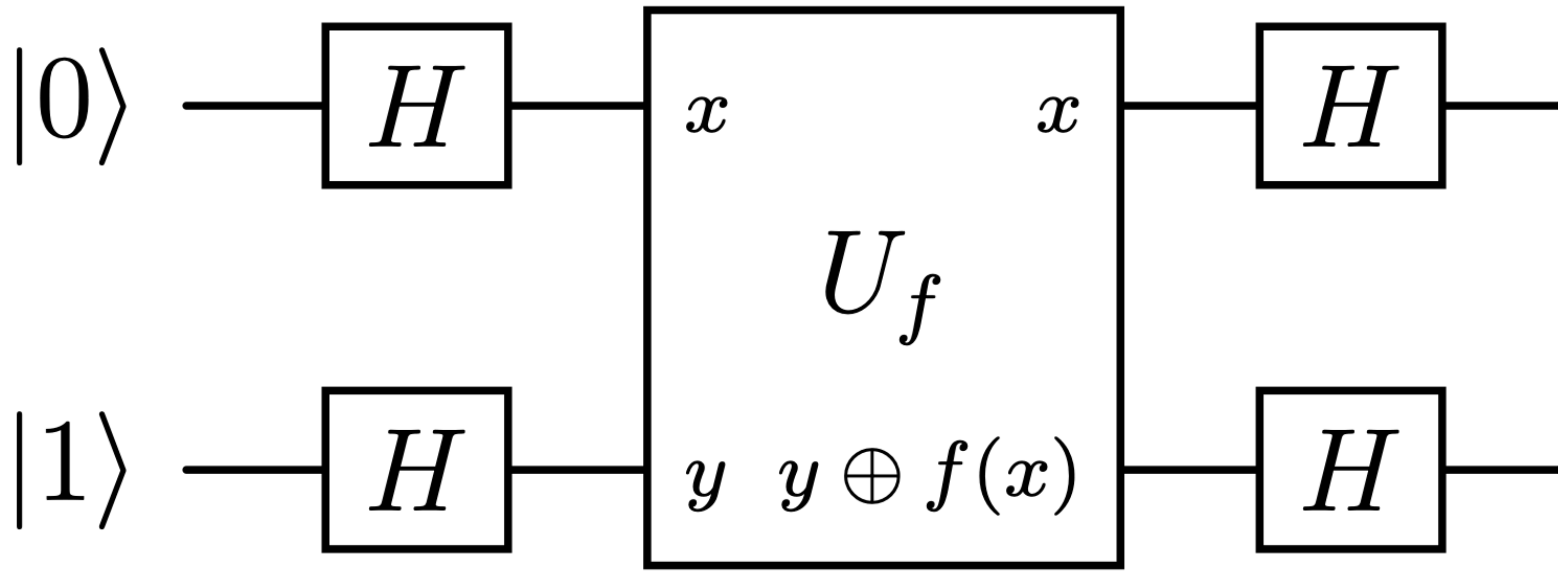
Hadamard operator

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

$$\frac{1}{2}\Big(\big((-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle\big) \otimes \big(|0\rangle - |1\rangle\big)\Big)$$

$$\xrightarrow{\hat{H} \otimes \hat{H}}$$

Deutsch-Jozsa Algorithm



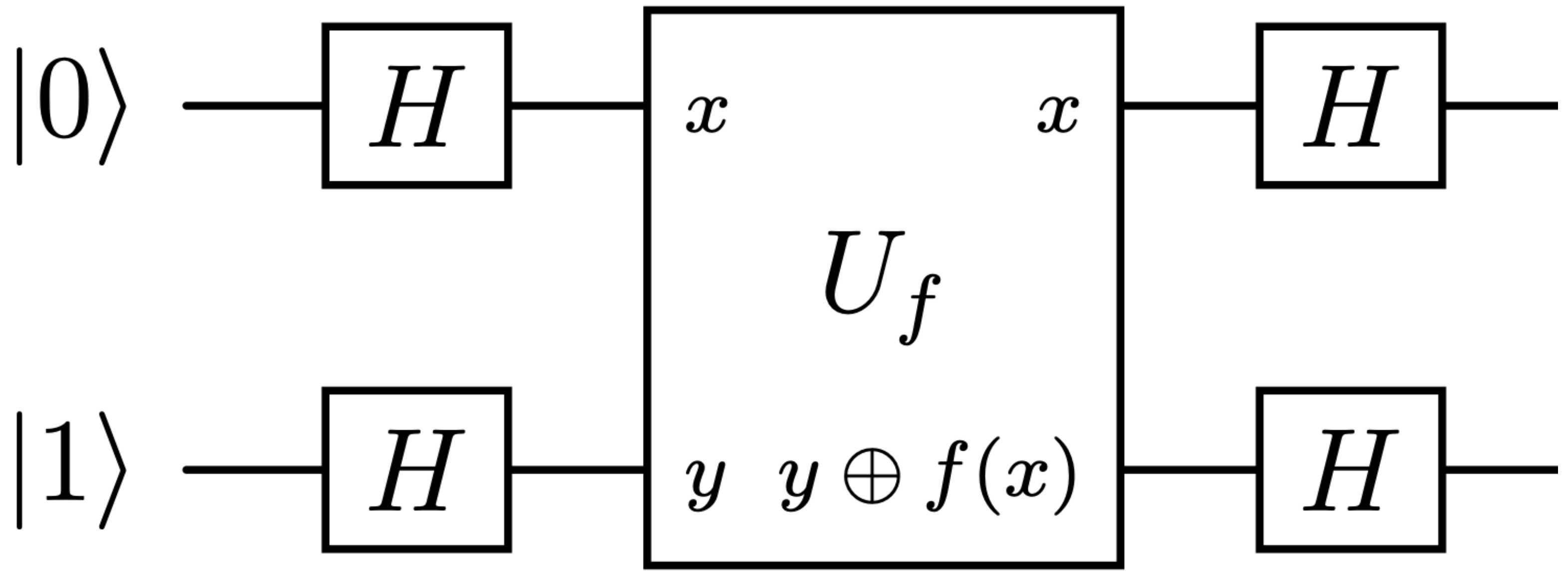
Hadamard operator

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

$$\frac{1}{2}\left(\left((-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle\right) \otimes (|0\rangle - |1\rangle)\right)$$

$$\xrightarrow{\hat{H}\otimes\hat{H}} \qquad \frac{1}{\sqrt{2}}\left(\hat{H}\left((-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle\right) \otimes |1\rangle\right)$$

Deutsch-Jozsa Algorithm



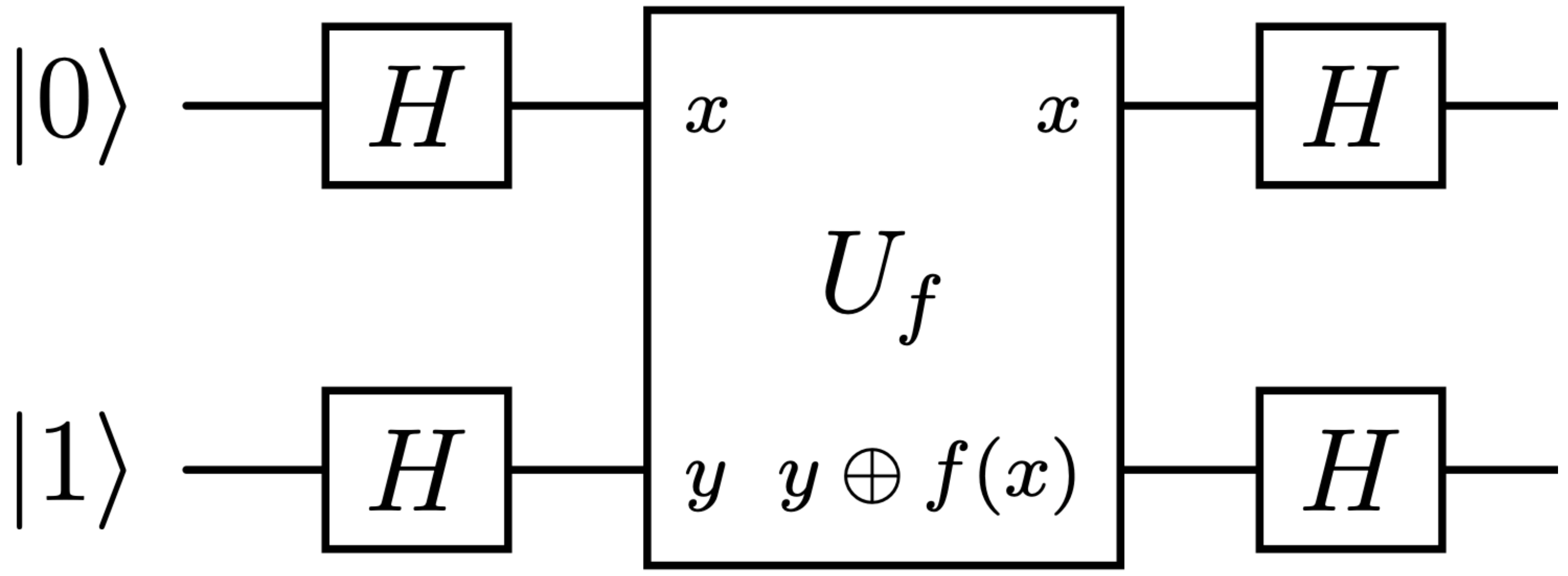
Hadamard operator

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

$$\frac{1}{\sqrt{2}}\left(\hat{H}((-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle) \otimes |1\rangle\right)$$

$$= \begin{cases} \dfrac{1}{\sqrt{2}}\left(\pm\hat{H}(|0\rangle + |1\rangle) \otimes |1\rangle\right) & \text{if} \quad f(0) = f(1) \\ \dfrac{1}{\sqrt{2}}\left(\pm\hat{H}(|0\rangle - |1\rangle) \otimes |1\rangle\right) & \text{if} \quad f(0) \neq f(1) \end{cases}$$

Deutsch-Jozsa Algorithm



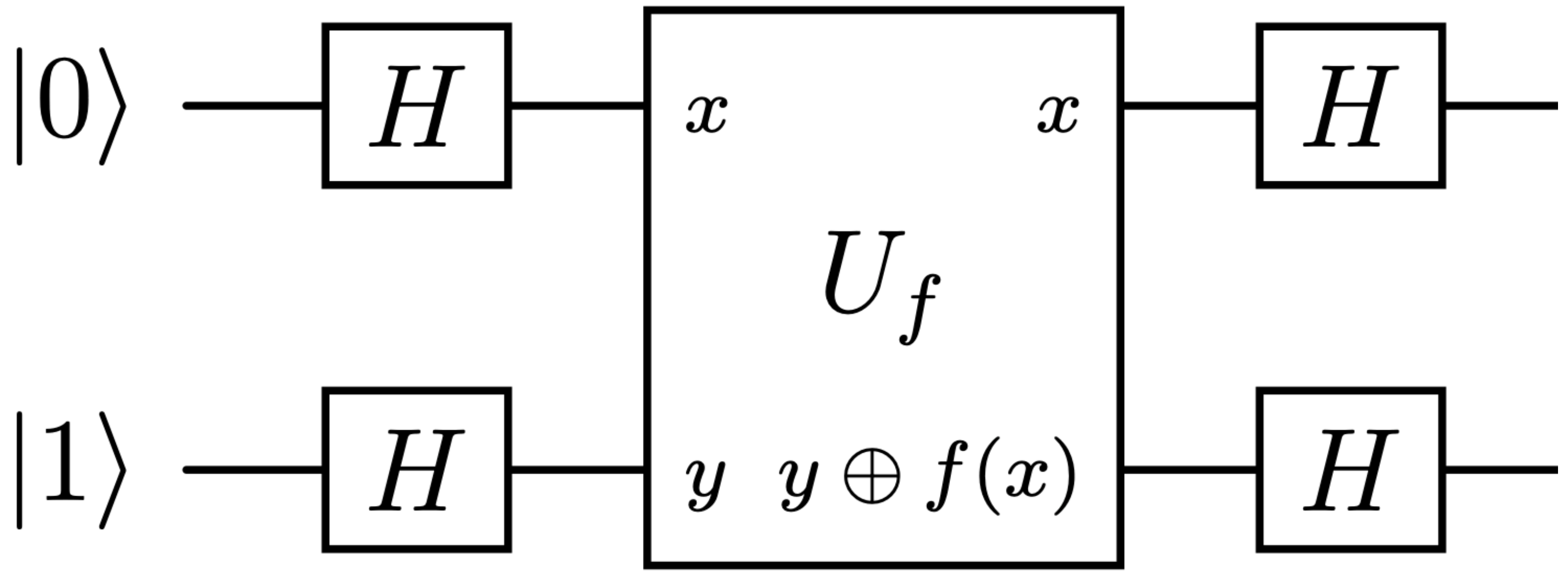
Hadamard operator

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

$$\frac{1}{\sqrt{2}}\left(\hat{H}((-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle) \otimes |1\rangle\right)$$

$$= \begin{cases} \dfrac{1}{\sqrt{2}}\left(\pm\sqrt{2}\,|0\rangle \otimes |1\rangle\right) & \text{if} \quad f(0) = f(1) \\ \dfrac{1}{\sqrt{2}}\left(\pm\hat{H}(|0\rangle - |1\rangle) \otimes |1\rangle\right) & \text{if} \quad f(0) \neq f(1) \end{cases}$$

Deutsch-Jozsa Algorithm



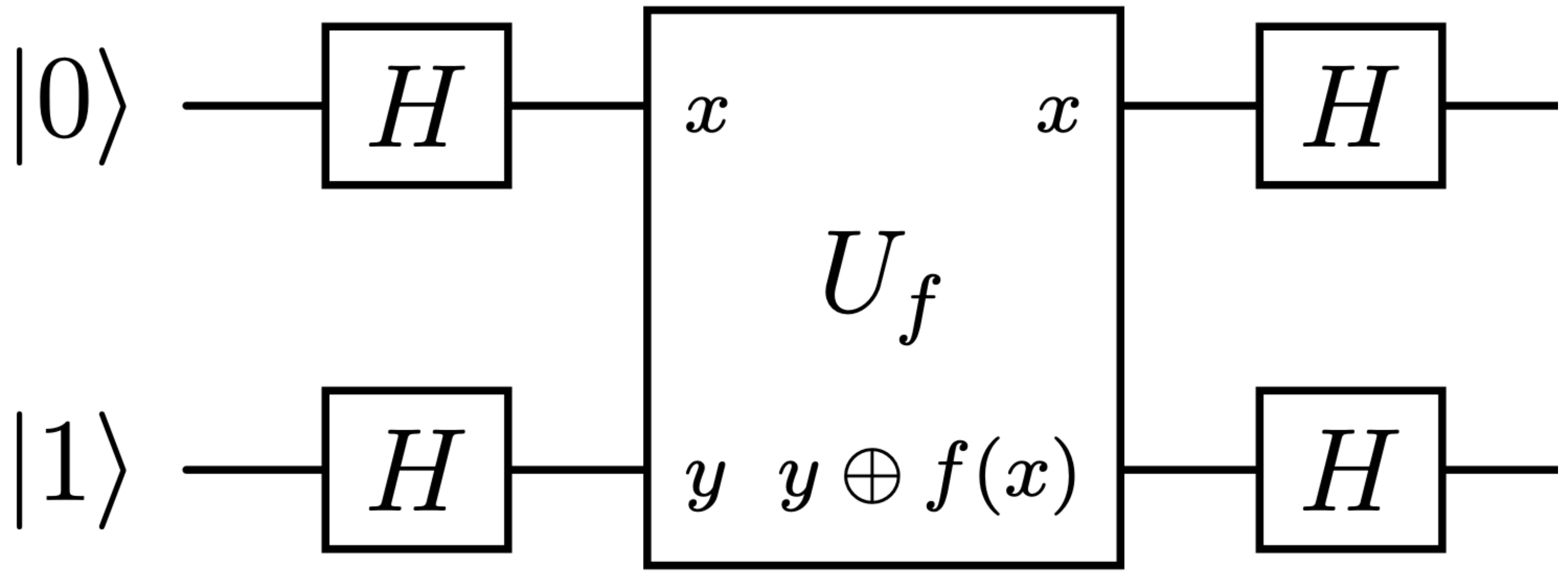
Hadamard operator

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

$$\frac{1}{\sqrt{2}}\left(\hat{H}\left((-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle\right) \otimes |1\rangle\right)$$

$$= \begin{cases} \dfrac{1}{\sqrt{2}}\left(\pm\sqrt{2}\,|0\rangle \otimes |1\rangle\right) & \text{if} \quad f(0) = f(1) \\ \dfrac{1}{\sqrt{2}}\left(\pm\sqrt{2}\,|1\rangle \otimes |1\rangle\right) & \text{if} \quad f(0) \neq f(1) \end{cases}$$

Deutsch-Jozsa Algorithm



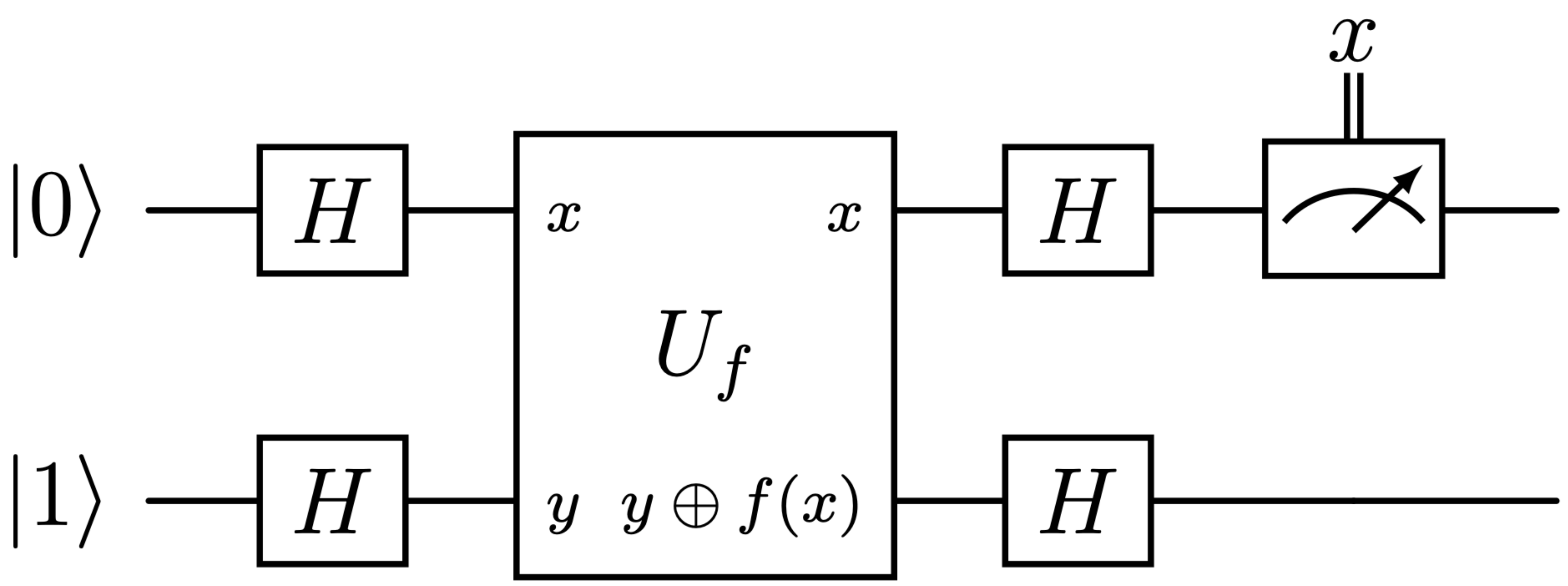
Hadamard operator

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

$$\frac{1}{\sqrt{2}}\left(\hat{H}\left((-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle\right) \otimes |1\rangle\right)$$

$$= \begin{cases} \pm|0\rangle \otimes |1\rangle & \text{if} \quad f(0) = f(1) \\ \pm|1\rangle \otimes |1\rangle & \text{if} \quad f(0) \neq f(1) \end{cases}$$

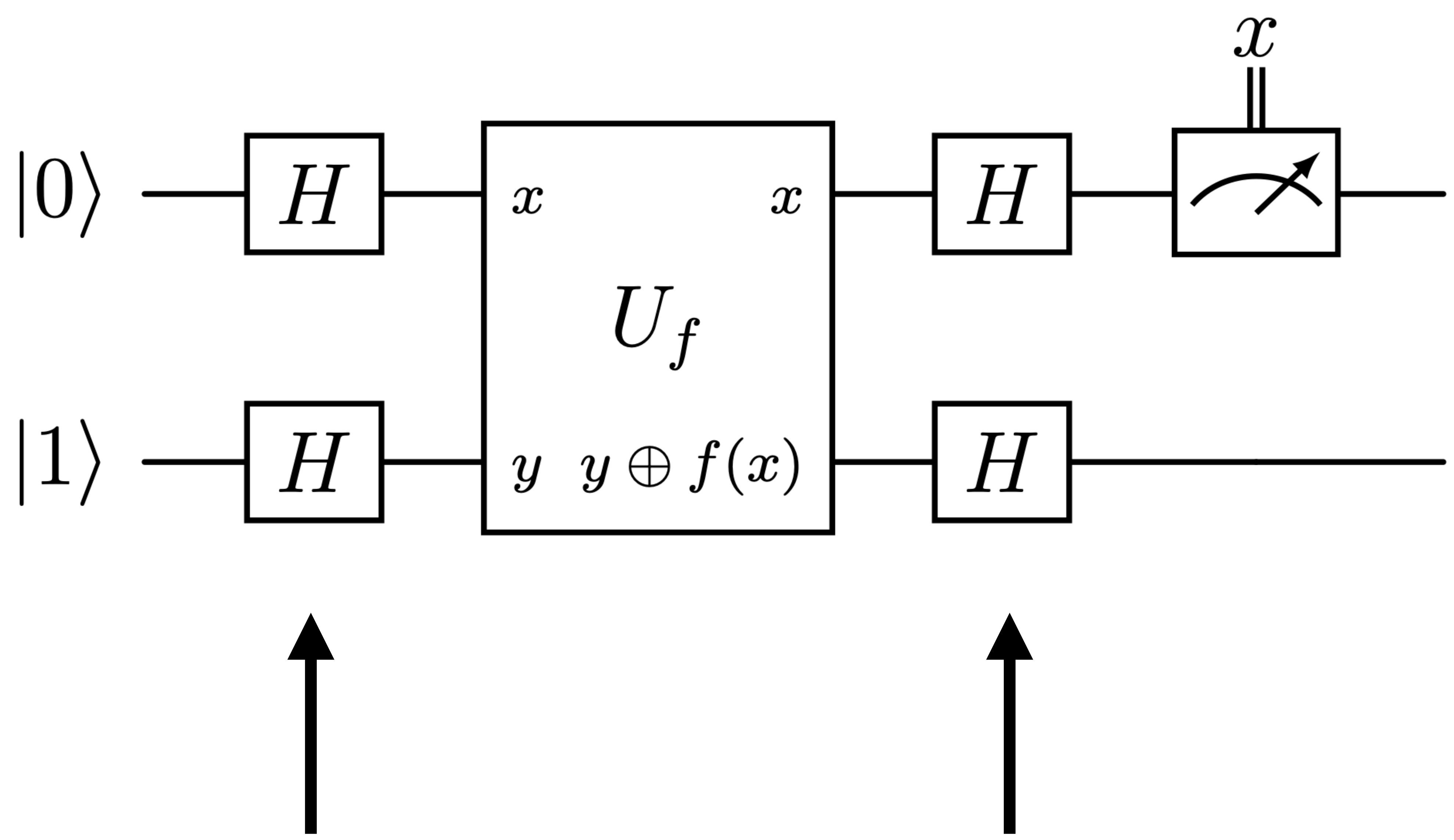
Deutsch-Jozsa Algorithm



$$\frac{1}{\sqrt{2}}\left(\hat{H}((-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle) \otimes |1\rangle\right)$$

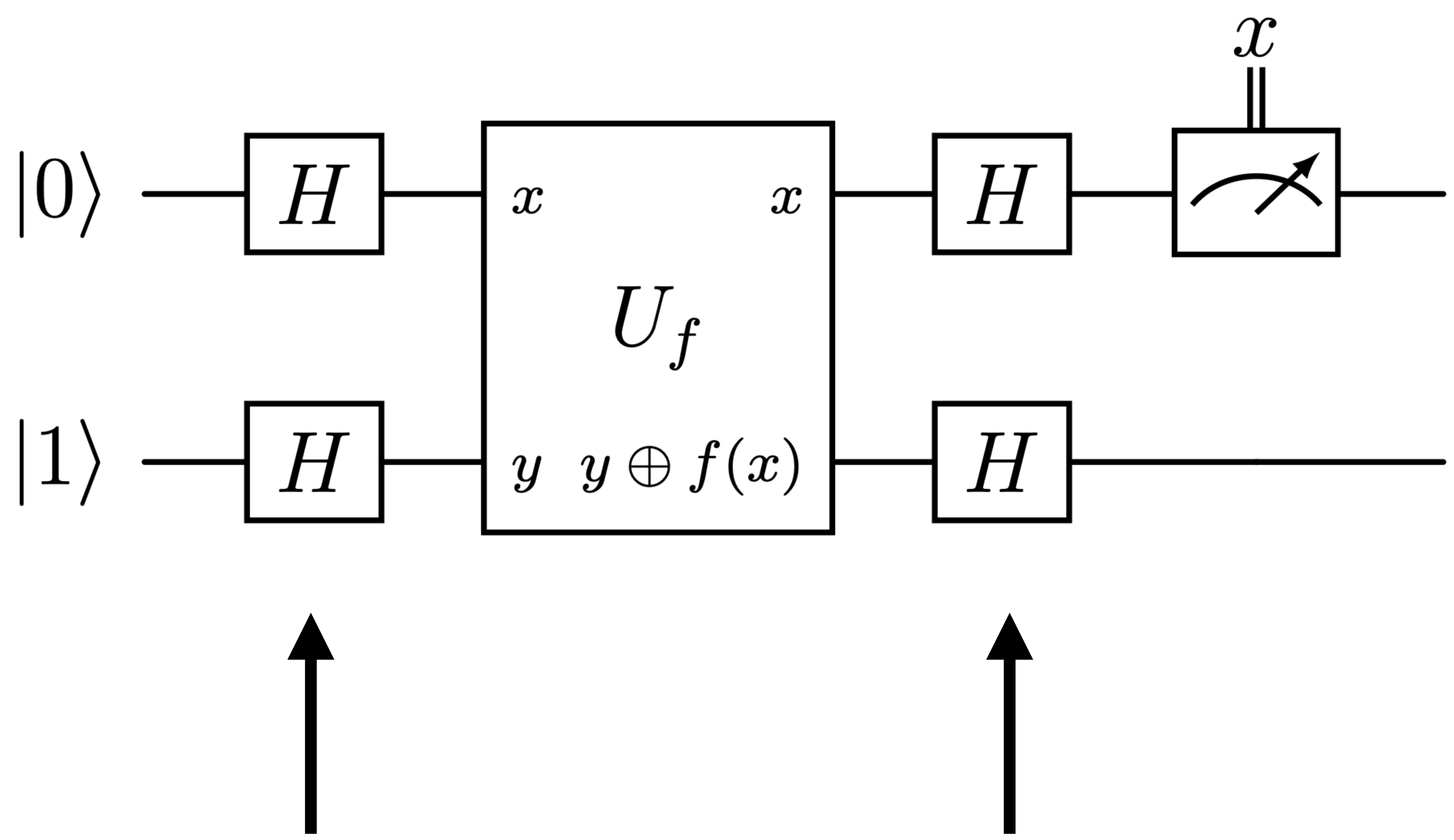
$$= \begin{cases} \pm\boxed{|0\rangle}\otimes|1\rangle & (x = 1) \quad \text{if} \quad f(0) = f(1) \\ \pm\boxed{|1\rangle}\otimes|1\rangle & (x = -1) \quad \text{if} \quad f(0) \neq f(1) \end{cases}$$

Deutsch-Jozsa Algorithm



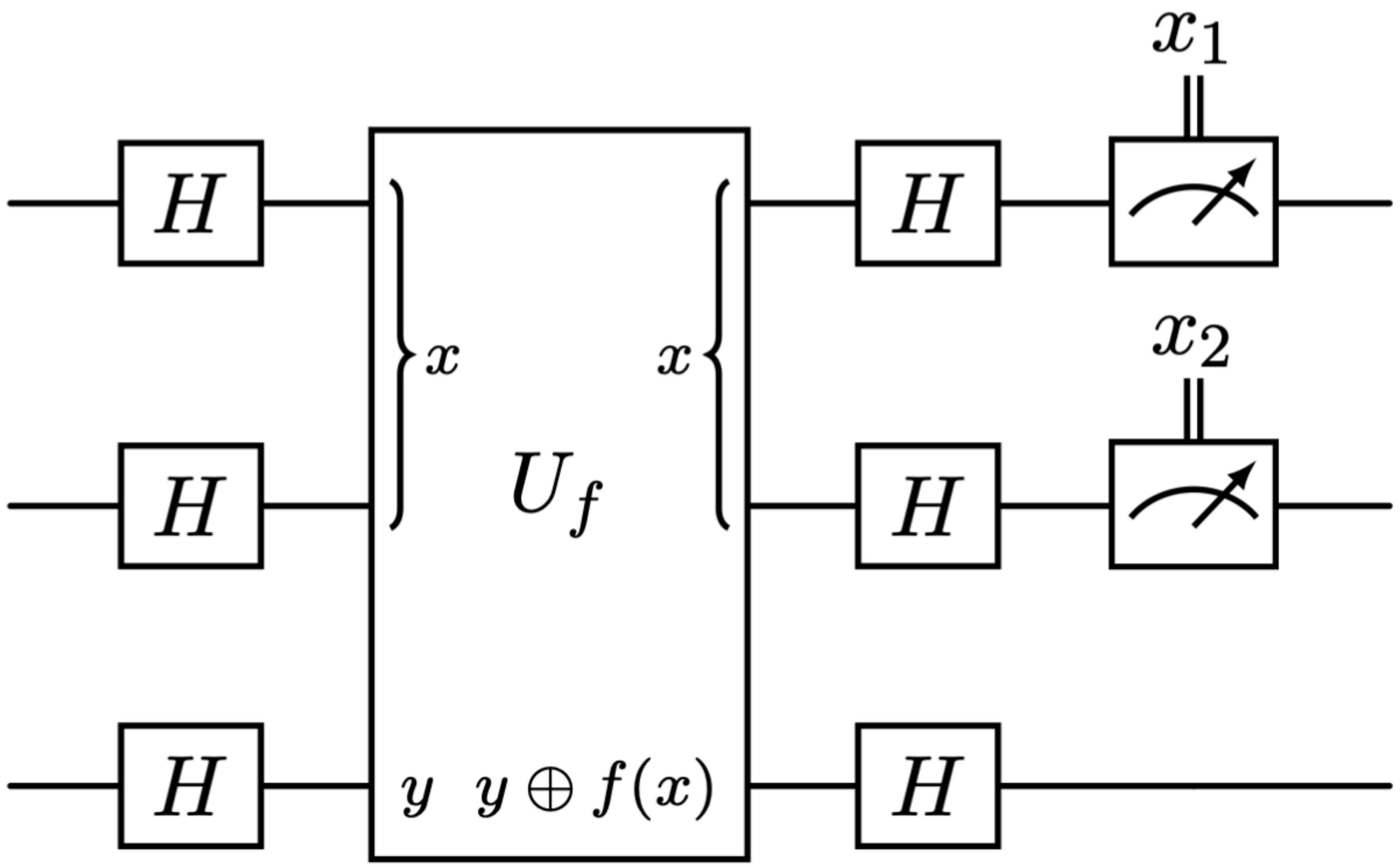
the Hadamard gates are the key *ingredient*

Deutsch-Jozsa Algorithm



the Hadamard gates are the key *ingredient* ⟶ apply f on a superposition of $0, 1$

Deutsch-Jozsa Algorithm



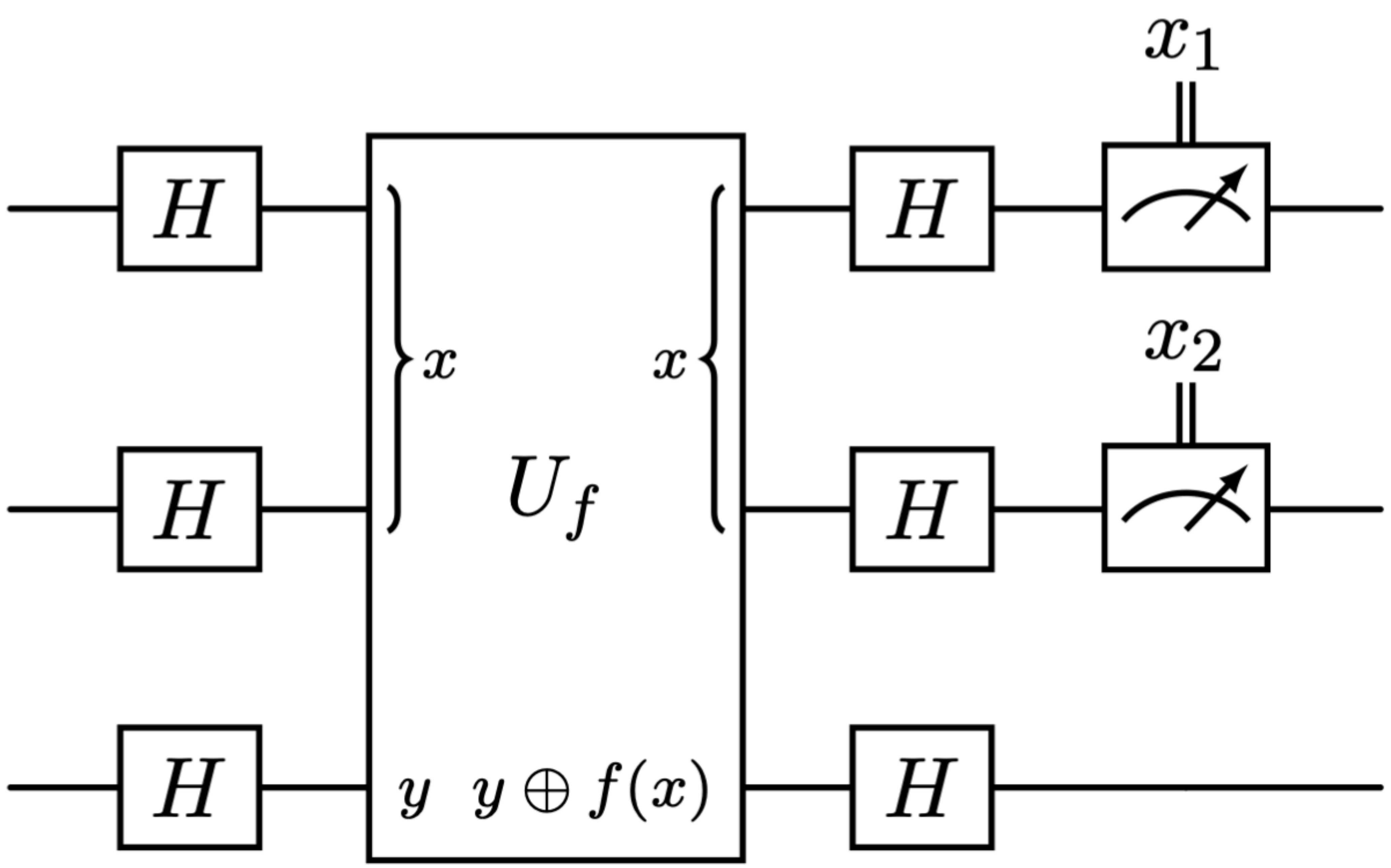
Algorithm extends to

$$f : \{0,1\}^n \to \{0,1\}$$

f is balanced if and only if

$$(x_1, x_2, \ldots, x_n) = (-1, -1, ..., -1)$$

Deutsch-Jozsa Algorithm



Algorithm extends to

$$f : \{0,1\}^n \to \{0,1\}$$

f is balanced if and only if

$$(x_1, x_2, \ldots, x_n) = (-1, -1, ..., -1)$$

Classical $2^{\frac{n}{2}} + 1$ ⟶ Quantum 1 step

Complexity Theory

Complexity Theory

What is the computational cost of an algorithm?

in space (memory) and time (runtime)

Complexity Theory

What is the computational cost of an algorithm?

in space (memory) and time (runtime)

Landau notation $$\mathcal{O}(f(n)) = \left\{ g : \mathbb{R} \to \mathbb{R} \,\middle|\, \limsup_{n\to\infty} \left| \frac{g(n)}{f(n)} \right| < \infty \right\}$$ set of upper bounds

Complexity Theory

What is the computational cost of an algorithm?

in space (memory) and time (runtime)

Landau notation $\mathcal{O}(f(n)) = \left\{ g : \mathbb{R} \to \mathbb{R} \,\middle|\, \limsup_{n\to\infty} \left| \frac{g(n)}{f(n)} \right| < \infty \right\}$ set of upper bounds

L $\mathcal{O}(\log n)$ space and time complexity

Complexity Theory

What is the computational cost of an algorithm?

in space (memory) and time (runtime)

Landau notation $\mathcal{O}(f(n)) = \left\{ g : \mathbb{R} \to \mathbb{R} \mid \limsup_{n\to\infty} \left| \frac{g(n)}{f(n)} \right| < \infty \right\}$ set of upper bounds

L $\mathcal{O}(\log n)$ space and time complexity

P $\mathcal{O}(p(n))$ space and time complexity

Complexity Theory

What is the computational cost of an algorithm?

in space (memory) and time (runtime)

Landau notation $\mathcal{O}(f(n)) = \left\{ g : \mathbb{R} \to \mathbb{R} \,|\, \limsup_{n\to\infty} \left| \frac{g(n)}{f(n)} \right| < \infty \right\}$ set of upper bounds

L $\mathcal{O}(\log n)$ space and time complexity

P $\mathcal{O}(p(n))$ space and time complexity

PSPACE $\mathcal{O}(p(n))$ space complexity

Complexity Theory

What is the computational cost of an algorithm?

in space (memory) and time (runtime)

Landau notation $\mathcal{O}(f(n)) = \left\{ g : \mathbb{R} \to \mathbb{R} \mid \limsup_{n\to\infty} \left| \frac{g(n)}{f(n)} \right| < \infty \right\}$ set of upper bounds

L	$\mathcal{O}(\log n)$ space and time complexity
P	$\mathcal{O}(p(n))$ space and time complexity
PSPACE	$\mathcal{O}(p(n))$ space complexity
EXP	$\mathcal{O}(2^{p(n)})$ space and time complexity

Complexity Theory

What is the computational cost of an algorithm?

in space (memory) and time (runtime)

Landau notation $\mathcal{O}(f(n)) = \left\{ g : \mathbb{R} \to \mathbb{R} \mid \limsup_{n\to\infty} \left| \frac{g(n)}{f(n)} \right| < \infty \right\}$ set of upper bounds

L $\mathcal{O}(\log n)$ space and time complexity

P $\mathcal{O}(p(n))$ space and time complexity

PSPACE $\mathcal{O}(p(n))$ space complexity

EXP $\mathcal{O}(2^{p(n)})$ space and time complexity

NP Solution verifiable in $\mathcal{O}(2^{p(n)})$ space and time

Complexity Theory

What is the computational cost of an algorithm?

in space (memory) and time (runtime)

Landau notation $\mathcal{O}(f(n)) = \left\{ g : \mathbb{R} \to \mathbb{R} \,\middle|\, \limsup_{n\to\infty} \left| \frac{g(n)}{f(n)} \right| < \infty \right\}$ set of upper bounds

L $\mathcal{O}(\log n)$ space and time complexity

P $\mathcal{O}(p(n))$ space and time complexity

PSPACE $\mathcal{O}(p(n))$ space complexity

EXP $\mathcal{O}(2^{p(n)})$ space and time complexity

NP Solution verifiable in $\mathcal{O}(p(n))$ space and time

BPP **P** with an bounded error probability of at least 2/3

Complexity Class NP

NP Solution verifiable in $\mathcal{O}(p(n))$ space and time

		3		2		6		
9			3		5			1
		1	8		6	4		
		8	1		2	9		
7								8
		6	7		8	2		
		2	6		9	5		
8			2		3			9
		5		1		3		

4	8	3	9	2	1	6	5	7
9	6	7	3	4	5	8	2	1
2	5	1	8	7	6	4	9	3
5	4	8	1	3	2	9	7	6
7	2	9	5	6	4	1	3	8
1	3	6	7	9	8	2	4	5
3	7	2	6	8	9	5	1	4
8	1	4	2	5	3	7	6	9
6	9	5	4	1	7	3	8	2

Complexity Class NP

NP Solution verifiable in $\mathcal{O}(p(n))$ space and time

		3		2		6		
9			3		5			1
		1	8		6	4		
		8	1		2	9		
7								8
		6	7		8	2		
		2	6		9	5		
8			2		3			9
		5		1		3		

solve →

in **EXP**

verify ←

in **P**

4	8	3	9	2	1	6	5	7
9	6	7	3	4	5	8	2	1
2	5	1	8	7	6	4	9	3
5	4	8	1	3	2	9	7	6
7	2	9	5	6	4	1	3	8
1	3	6	7	9	8	2	4	5
3	7	2	6	8	9	5	1	4
8	1	4	2	5	3	7	6	9
6	9	5	4	1	7	3	8	2

where n is the number of empty squares

Complexity Theory

We know that

$\mathbf{L} \subsetneq \mathbf{PSPACE}$

$\mathbf{P} \subsetneq \mathbf{EXP}$

thus

$\mathbf{L} \subseteq \mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{PSPACE} \subseteq \mathbf{EXP}$

Complexity Theory

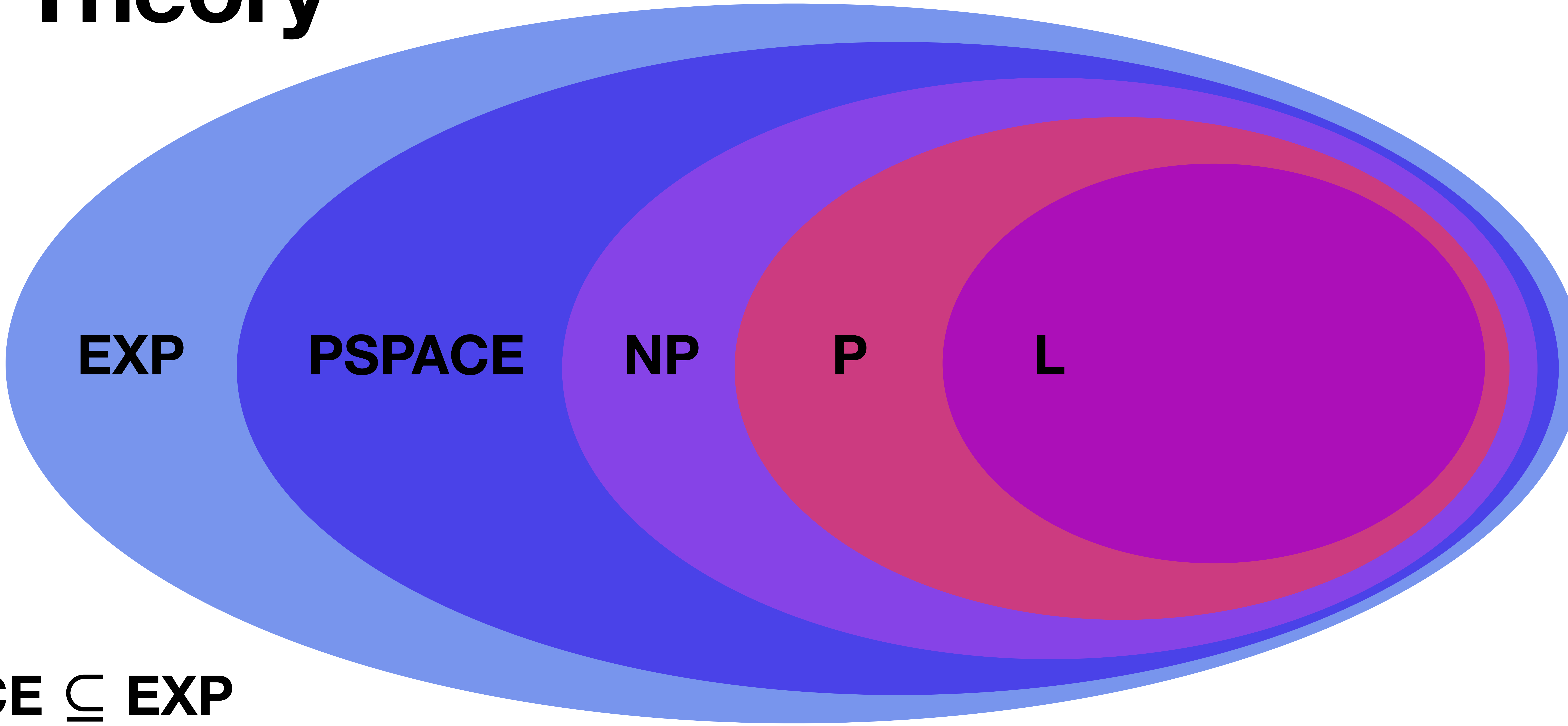
We know that

L $\subsetneq$ **PSPACE**

P $\subsetneq$ **EXP**

thus

L $\subseteq$ **P** $\subseteq$ **NP** $\subseteq$ **PSPACE** $\subseteq$ **EXP**



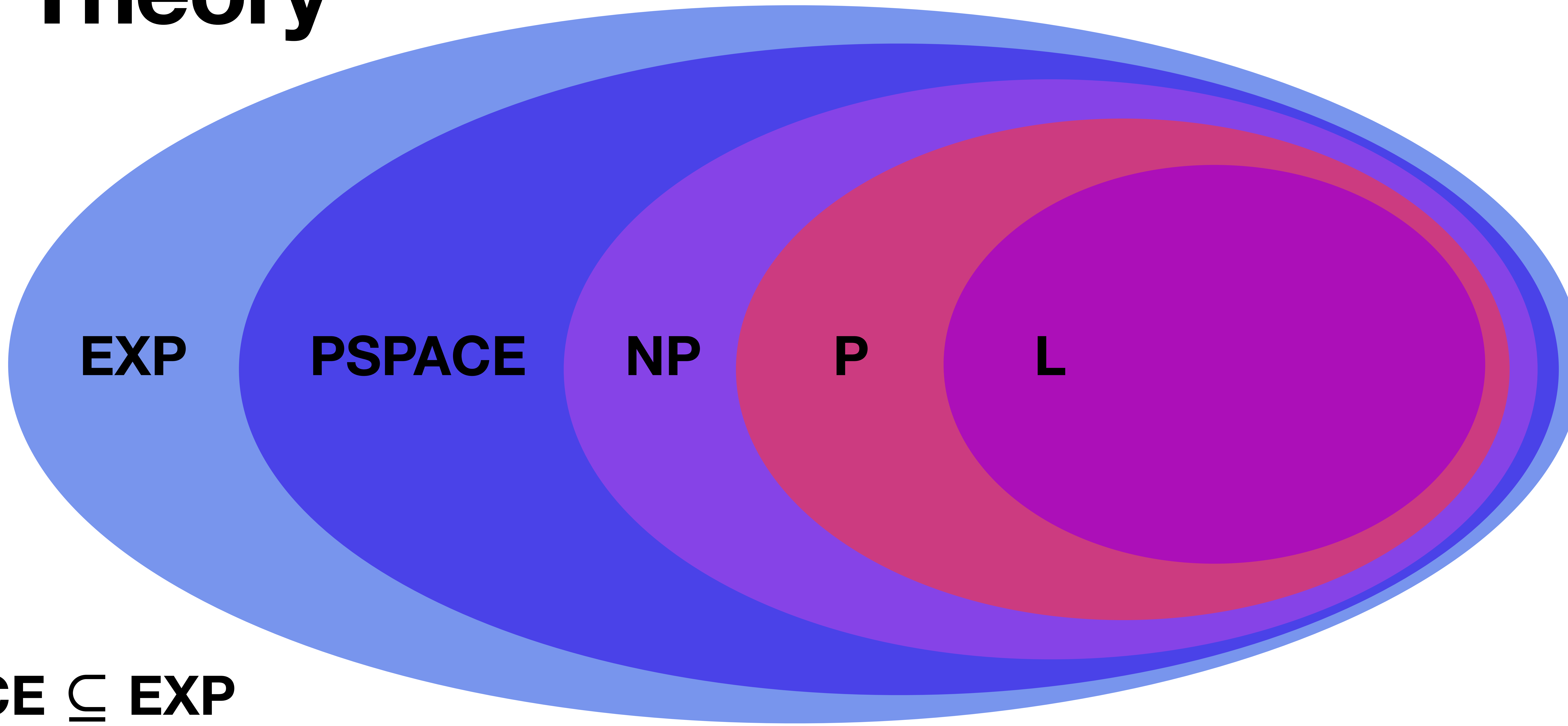
Complexity Theory

We know that

L $\subsetneq$ **PSPACE**

P $\subsetneq$ **EXP**

thus



L $\subseteq$ **P** $\subseteq$ **NP** $\subseteq$ **PSPACE** $\subseteq$ **EXP**

At least one of the inclusions needs to be strict → Open problem

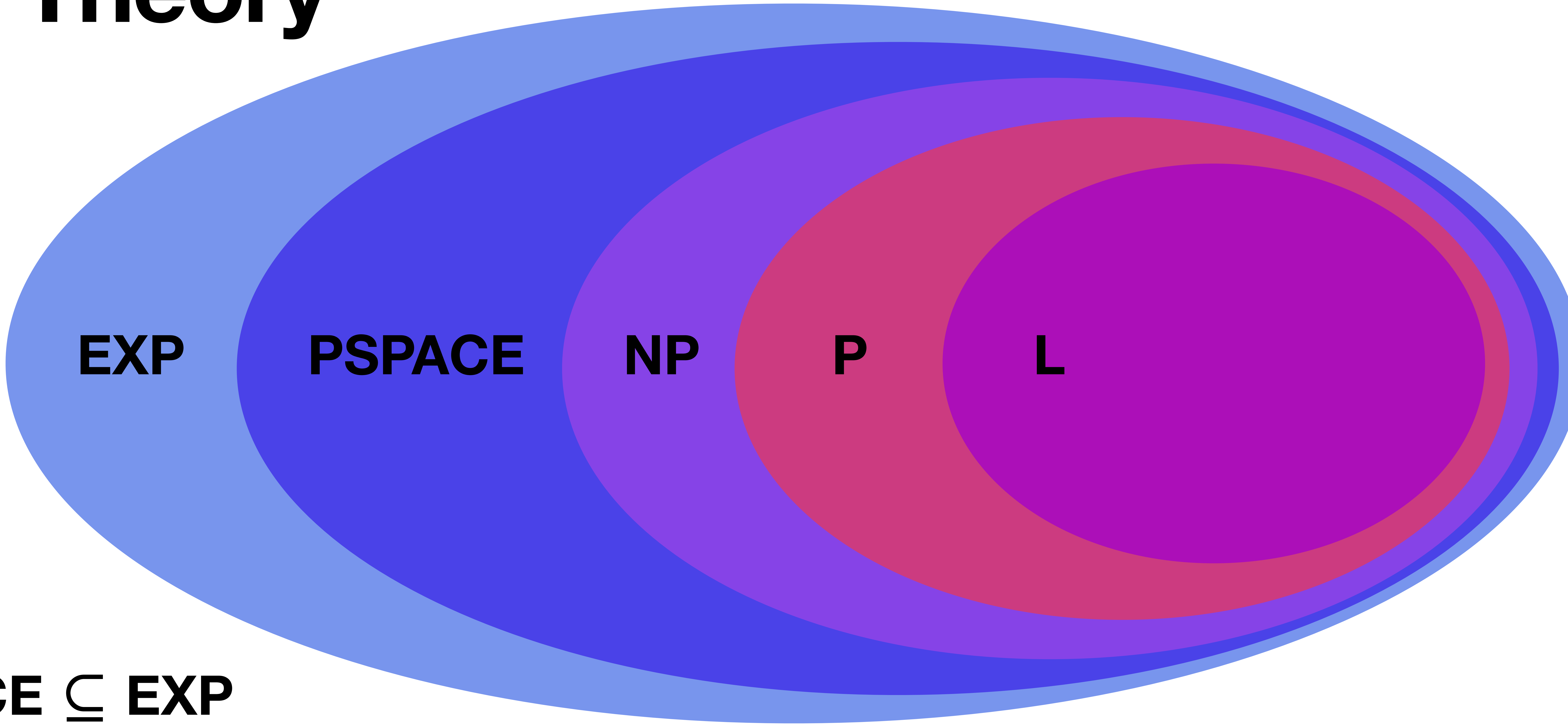
Complexity Theory

We know that

$\mathbf{L} \subsetneq \mathbf{PSPACE}$

$\mathbf{P} \subsetneq \mathbf{EXP}$

thus



$\mathbf{L} \subseteq \mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{PSPACE} \subseteq \mathbf{EXP}$

At least one of the inclusions needs to be strict → Open problem

$\boxed{\mathbf{NP} \stackrel{?}{=} \mathbf{P}}$ Millenium Problem

Quantum complexity classes

BQP $\mathcal{O}(p(n))$ space and time complexity on quantum computer

Bounded error Quantum Probability

Quantum equivalent to BPP

Quantum complexity classes

BQP $\mathcal{O}(p(n))$ space and time complexity on quantum computer

Bounded error Quantum Probability

Quantum equivalent to BPP

QMA Quantum proof verification with bounded error

Quantum Merlin Arthur

Quantum equivalent to NP

Quantum complexity classes

BQP $\mathcal{O}(p(n))$ space and time complexity on quantum computer

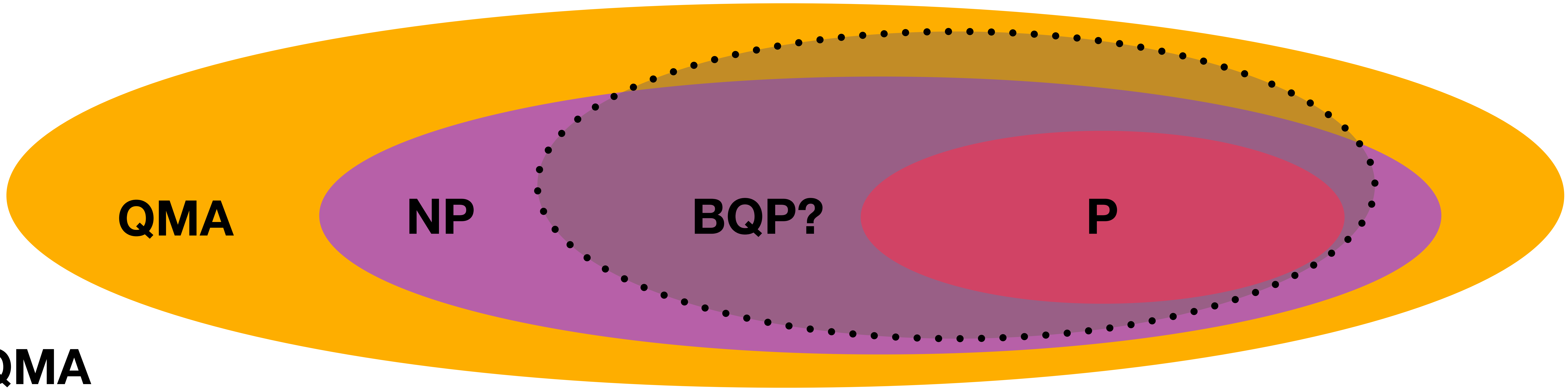
Bounded error Quantum Probability

Quantum equivalent to BPP

QMA Quantum proof verification with bounded error

Quantum Merlin Arthur

Quantum equivalent to NP



We know that

P $\subseteq$ BQP $\subseteq$ QMA
NP $\subseteq$ QMA

Quantum complexity classes

BQP $\mathcal{O}(p(n))$ space and time complexity on quantum computer

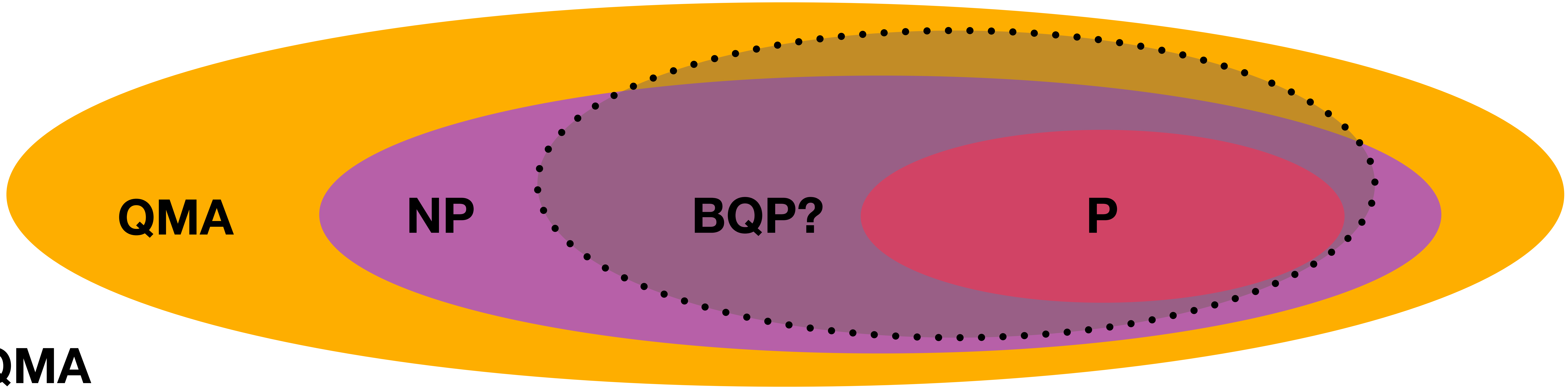
Bounded error Quantum Probability

Quantum equivalent to BPP

QMA Quantum proof verification with bounded error

Quantum Merlin Arthur

Quantum equivalent to NP



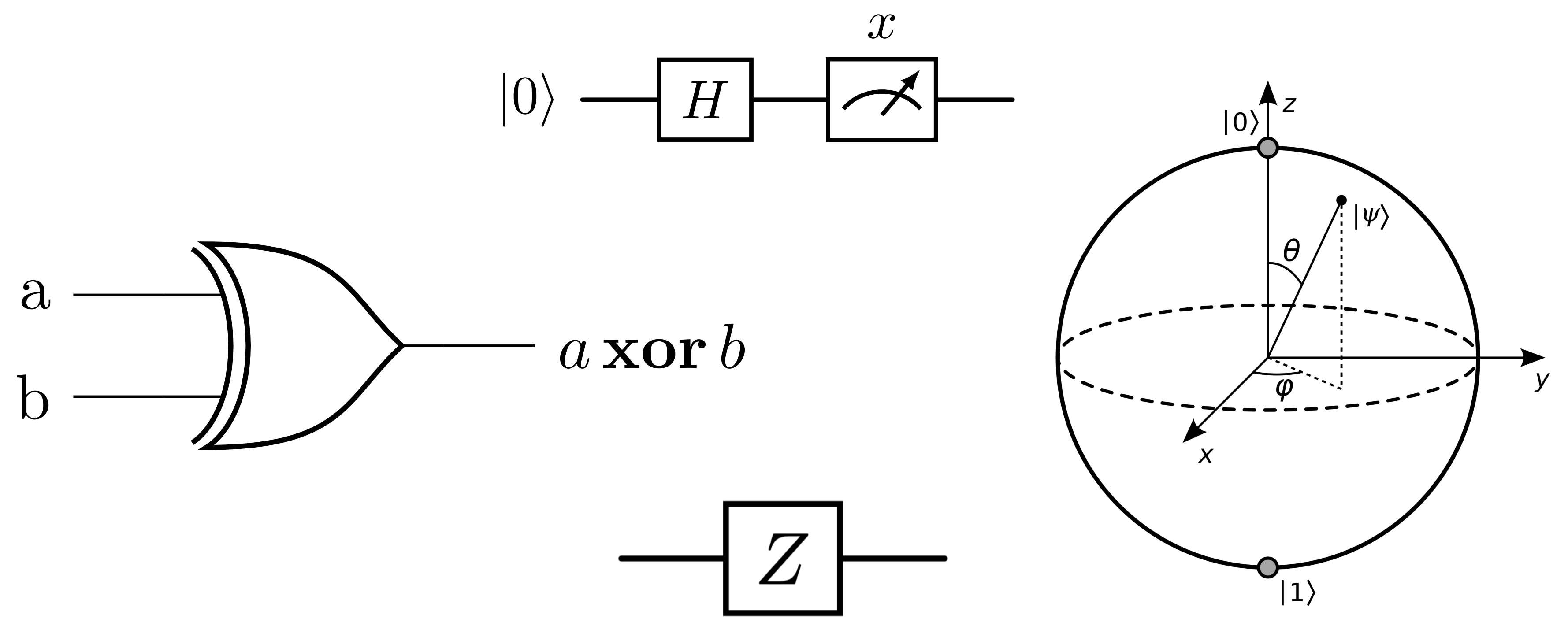
We know that

P $\subseteq$ BQP $\subseteq$ QMA
NP $\subseteq$ QMA

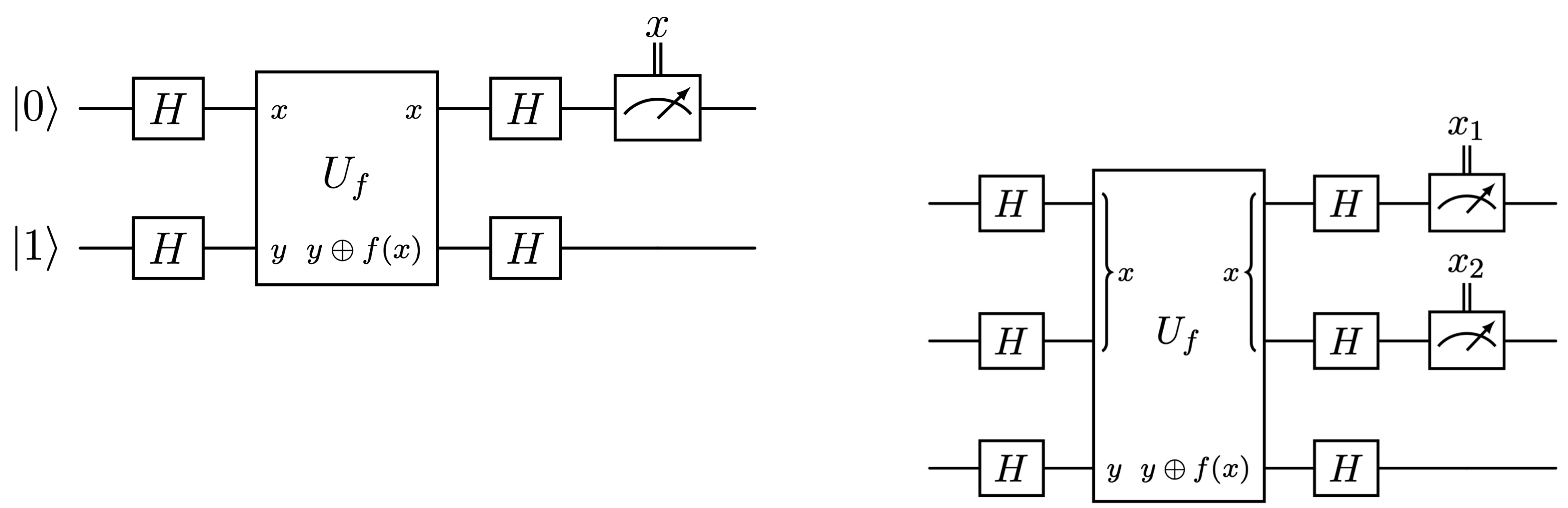
Open problem $\boxed{\textbf{NP} \overset{?}{\subseteq} \textbf{BQP}}$

Summary

Classical and Quantum Circuits

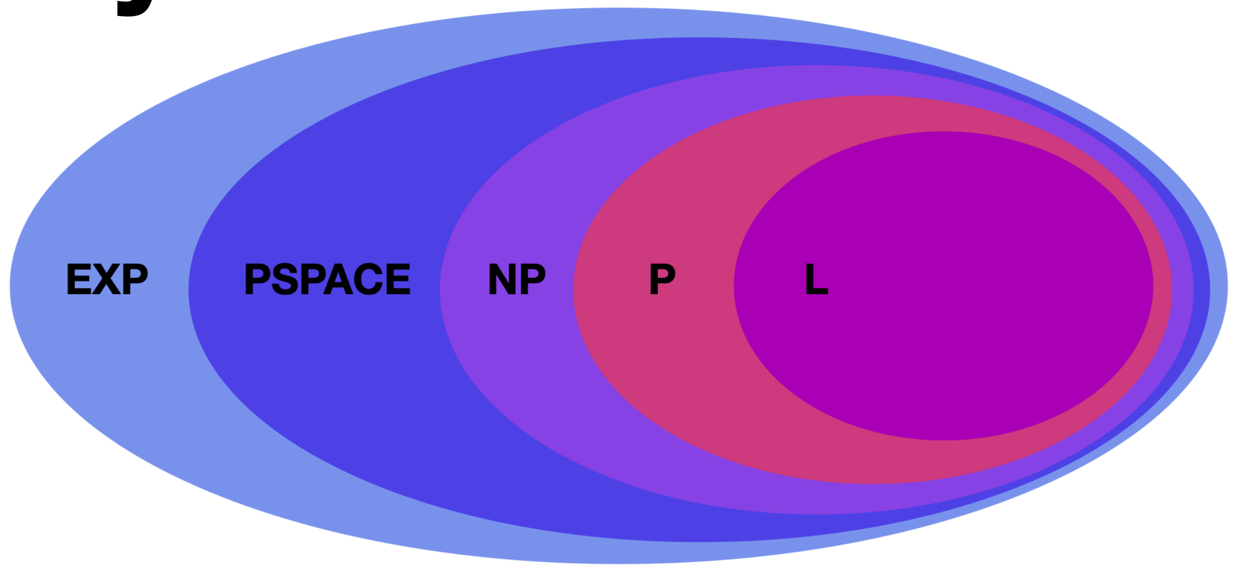


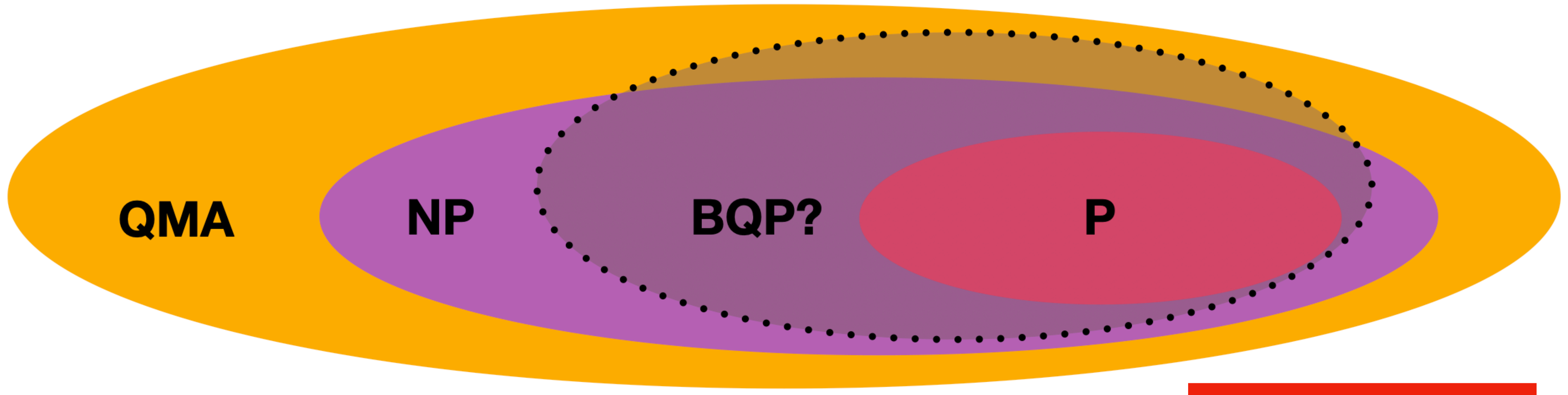
Quantum Algorithms



Complexity theory

$\textbf{NP} \stackrel{?}{=} \textbf{P}$



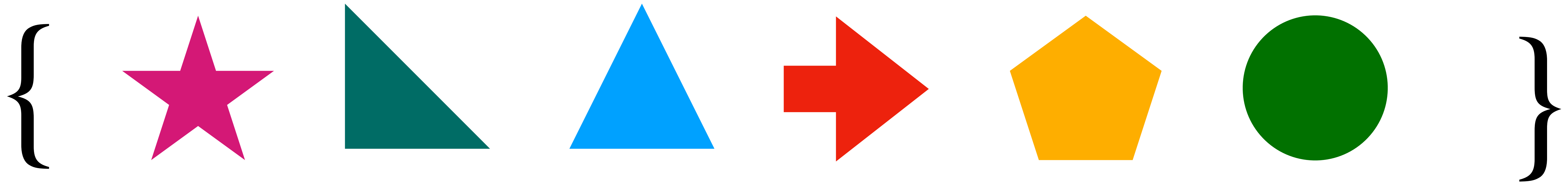


$\textbf{NP} \stackrel{?}{\subseteq} \textbf{BQP}$

Grover’s Algorithm

Classical Search

Searching an item in an unordered list of size N



takes on average $\dfrac{N}{2}$ steps $\rightarrow \mathcal{O}(N)$ in time, thus the problem is in **P**

Quantum Grover's Algorithm is $\mathcal{O}(\sqrt{N})$ in time $\rightarrow$ in **BQP**

Grover's Algorithm

We consider $N = 2$

Solution direction

Initial state $|s\rangle = \frac{1}{\sqrt{2}}(|\omega\rangle + |s'\rangle)$

1. step: $|s\rangle \rightarrow U_\omega |s\rangle$ $U_\omega = \mathbf{Id} - 2|\omega\rangle\langle\omega|$

Reflect the $|\omega\rangle$ - component

2. step: $U_\omega |s\rangle \rightarrow U_s U_\omega |s\rangle$ $U_s = 2|s\rangle\langle s| - \mathbf{Id}$

Reflect around $|s\rangle$ - component