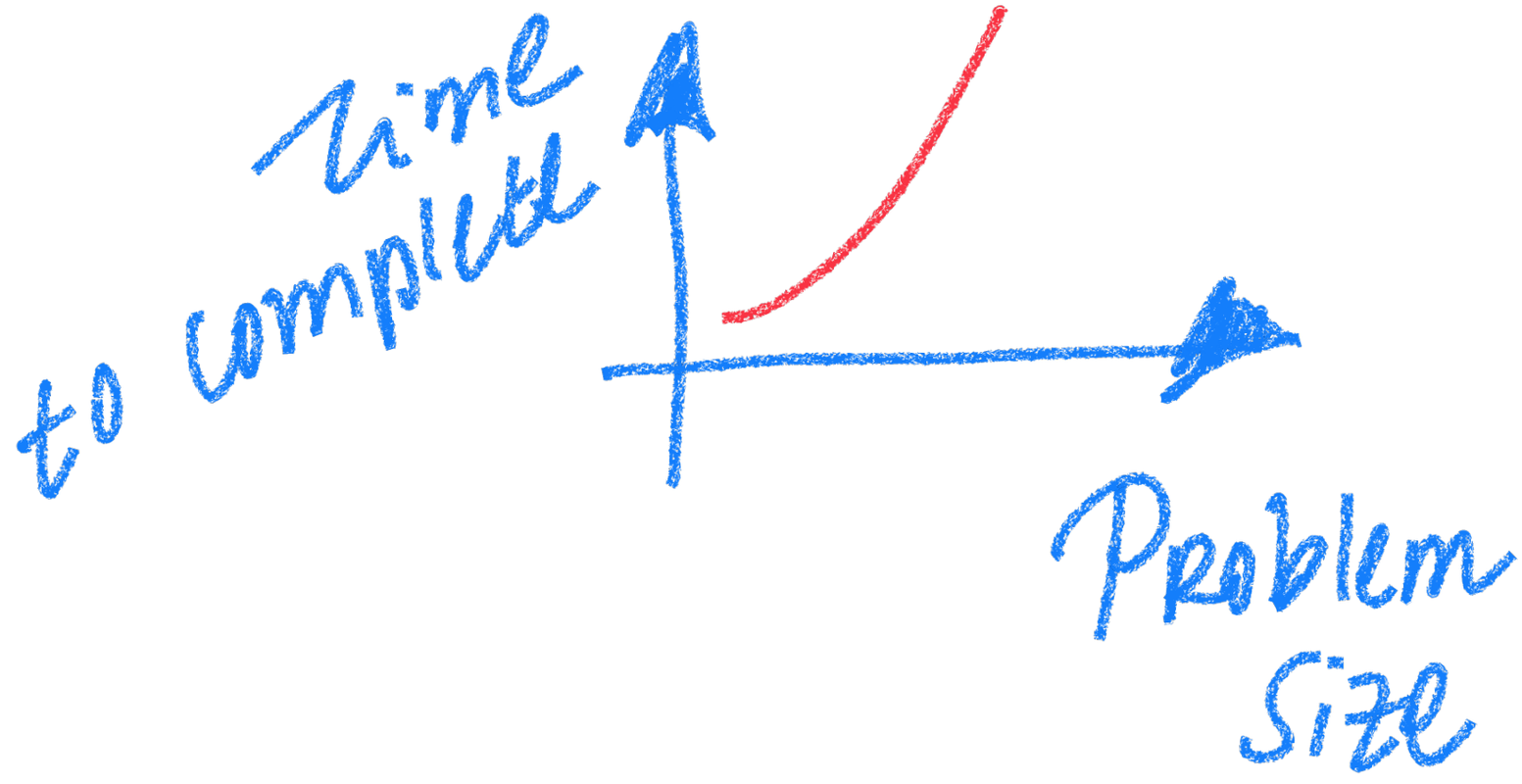
EE5340

INTRODUCTION TO QUANTUM COMPUTING AND PHYSICAL BASICS OF COMPUTING

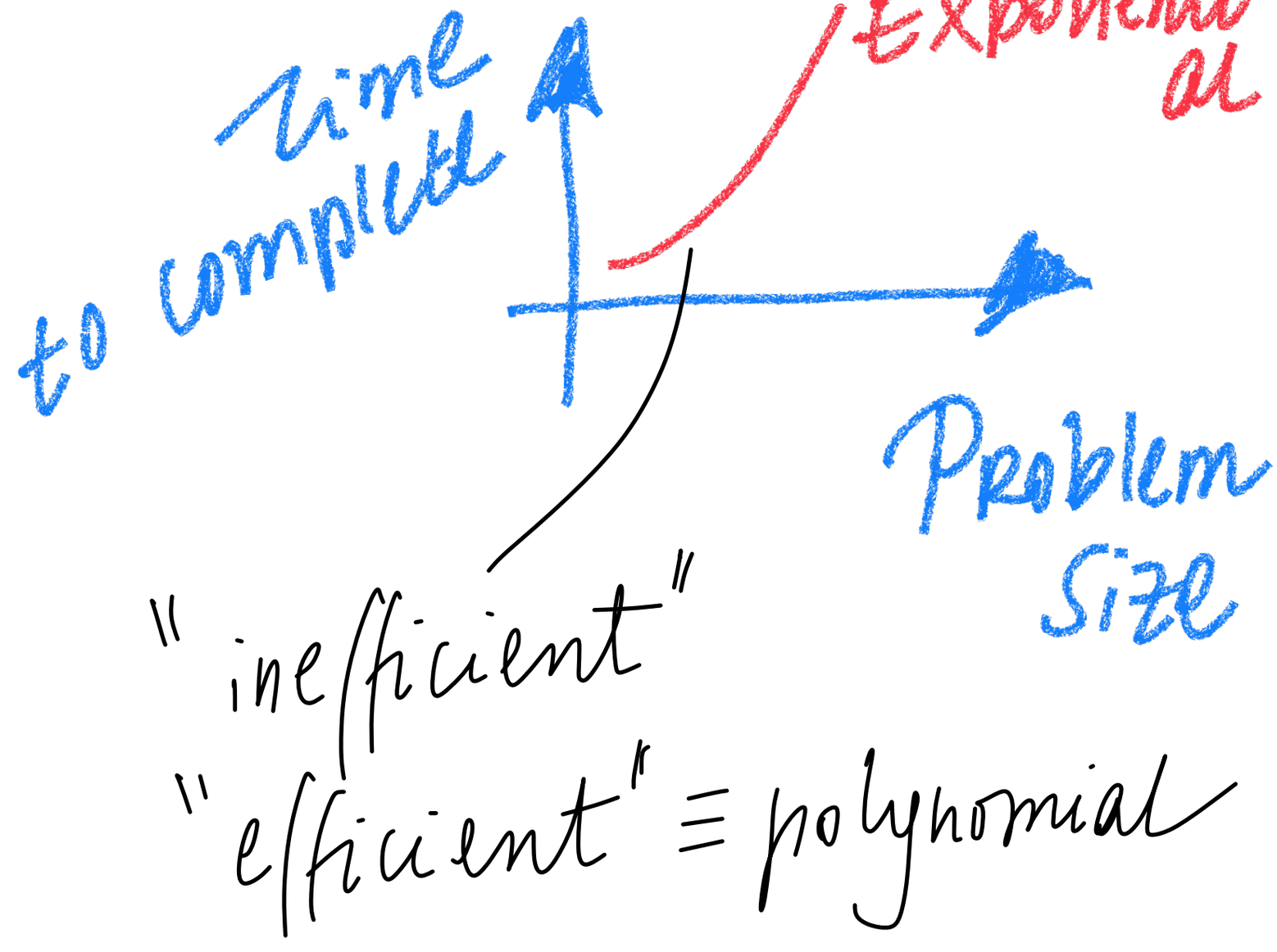
Quantum Algorithms

Ulya Karpuzcu

Functionality vs. Efficiency



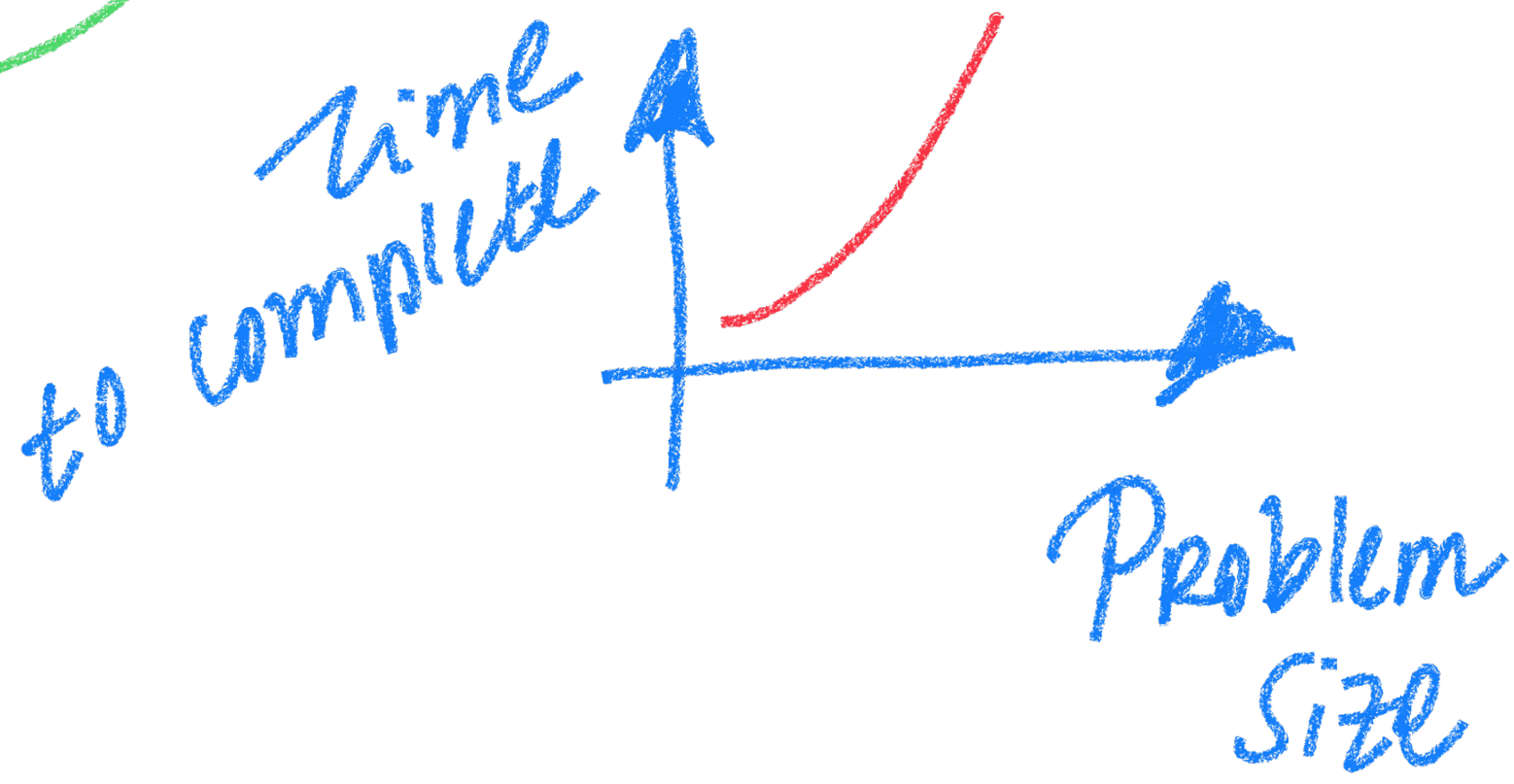
Functionality vs. Efficiency



"inefficient"

"efficient" ≡ polynomial

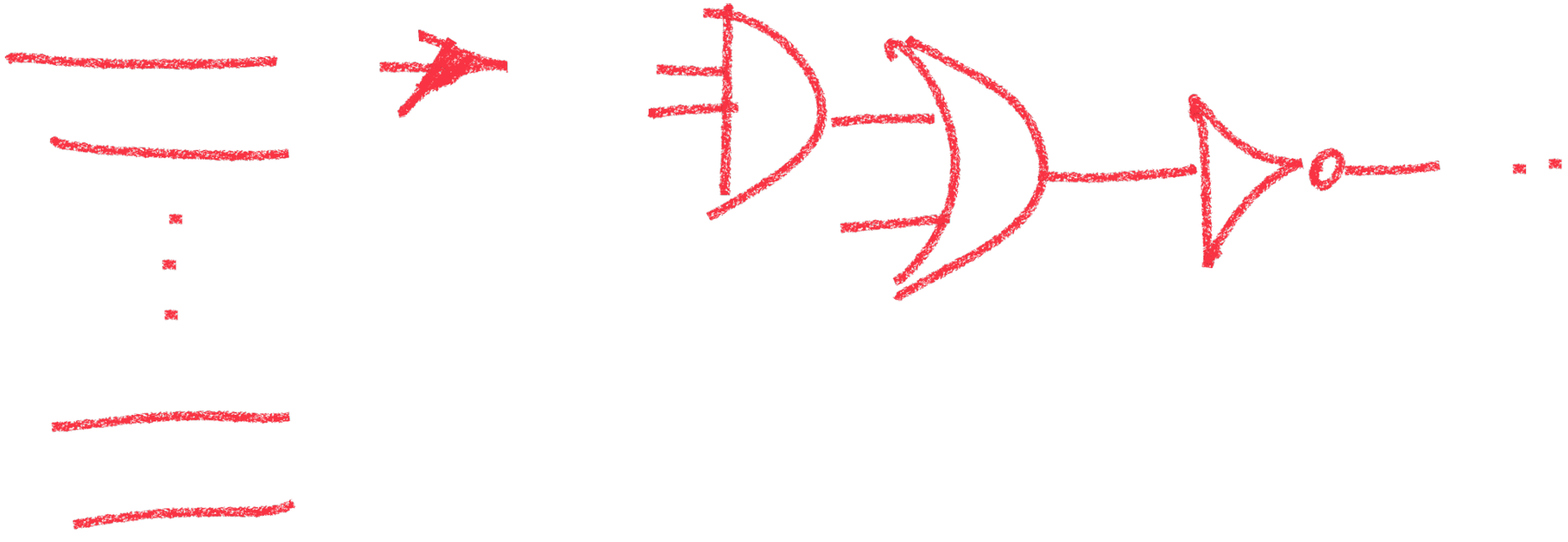
Functionality vs. Efficiency



Classic Computing on Quantum Computers

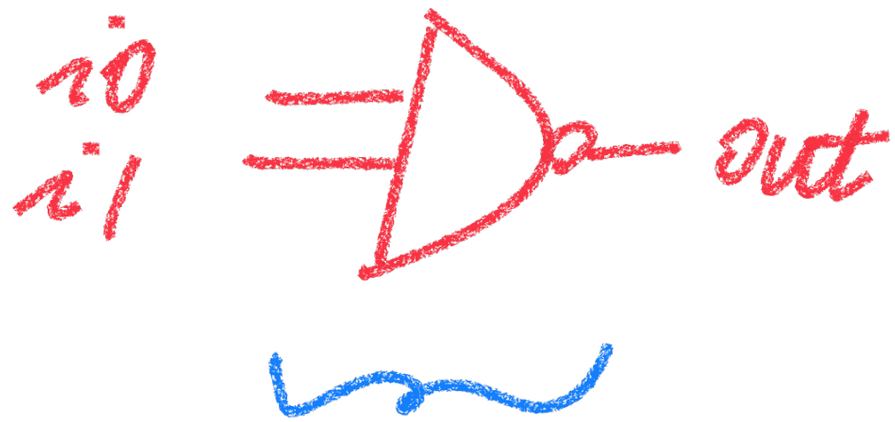
- Replace all logic by an equivalent, consisting of reversible elements
  - CCN(Toffoli) gate
    - Can mimic NAND and FANOUT
  - Quantum CCN?

Classical Alg.



NAND

i_0	i_1	out
0	0	1
0	1	1
1	0	1
1	1	0



Δt

vs.

Ex: $\mu t = 1 \Rightarrow i_0 = 2$
$i_1 = 2$

must be unitary!

$$G \cdot |input\rangle = |output\rangle$$

$$\langle input | input \rangle = 1$$

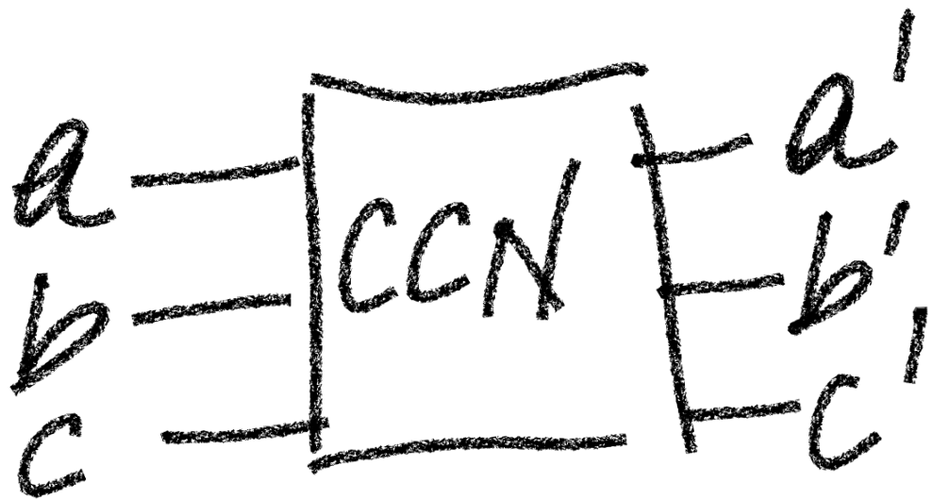
$$\langle output | output \rangle = 1$$

Toffoli Gate (CCN)

- $(a,b,c) \rightarrow (a,b,c \oplus ab)$
- Smallest universal reversible (classic) operation
  - Functionally complete (can mimic a NAND gate)

a	b	c	a'	b'	c'
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

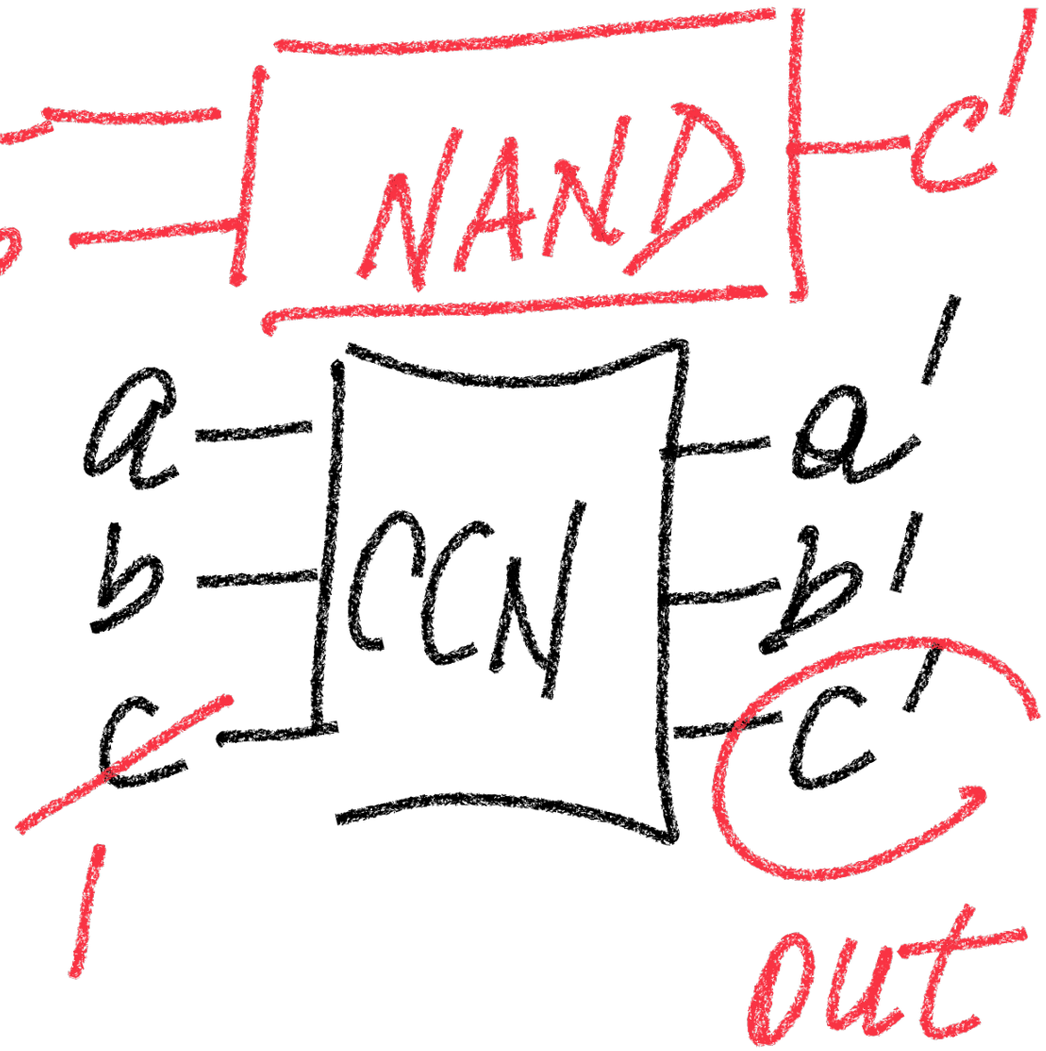
$a \wedge b = 1$



Toffoli Gate (CCN)

- (a,b,c) → (a,b,c ⊕ ab)
- Smallest universal reversible (classic) operation
  - NAND: c =1, read c'

a	b	c	a'	b'	c'
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0



c = 1

NAND

i0	i1	out
0	0	1
0	1	1
1	0	1
1	1	0

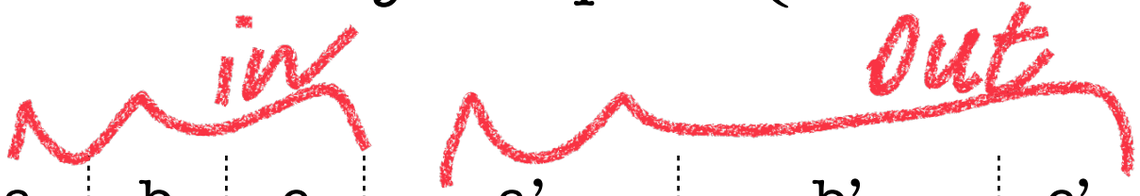
Toffoli Gate (CCN)

- (a,b,c) → (a,b,c ⊕ ab)
- Smallest universal reversible (classic) operation
  - FANOUT: a=1, c=0, b'=c'=b (a'=1)

a	b	c	a'	b'	c'
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

Toffoli Gate (CCN)

- (a,b,c) → (a,b,c ⊕ ab)
- Smallest universal reversible (classic) operation
  - Functionally complete (can mimic a NAND gate)

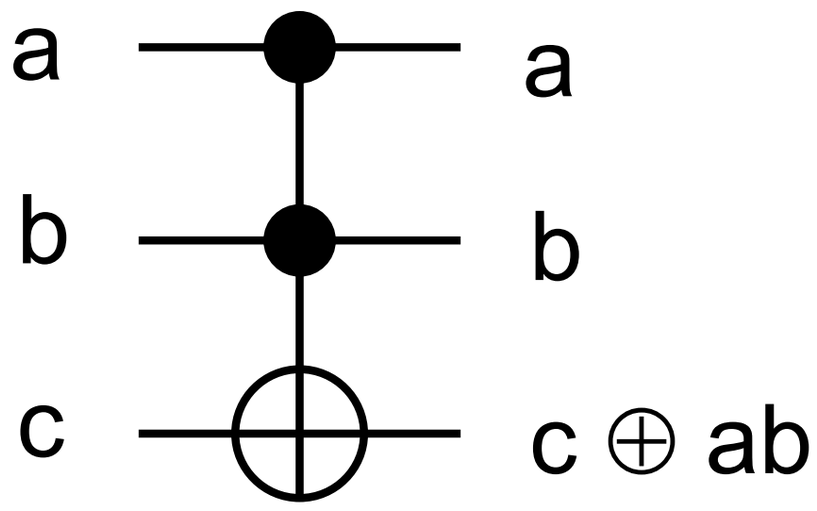


a	b	c	a'	b'	c'
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

$|000\rangle \rightarrow |000\rangle$; $|001\rangle \rightarrow |001\rangle$; $|010\rangle \rightarrow |010\rangle$; $|011\rangle \rightarrow |011\rangle$
$|100\rangle \rightarrow |100\rangle$; $|101\rangle \rightarrow |101\rangle$; $|110\rangle \rightarrow |111\rangle$; $|111\rangle \rightarrow |110\rangle$

Toffoli Gate (CCN)

- (a,b,c) → (a,b,c ⊕ ab)
- Smallest universal reversible (classic) operation
  - Functionally complete (can mimic a NAND gate)

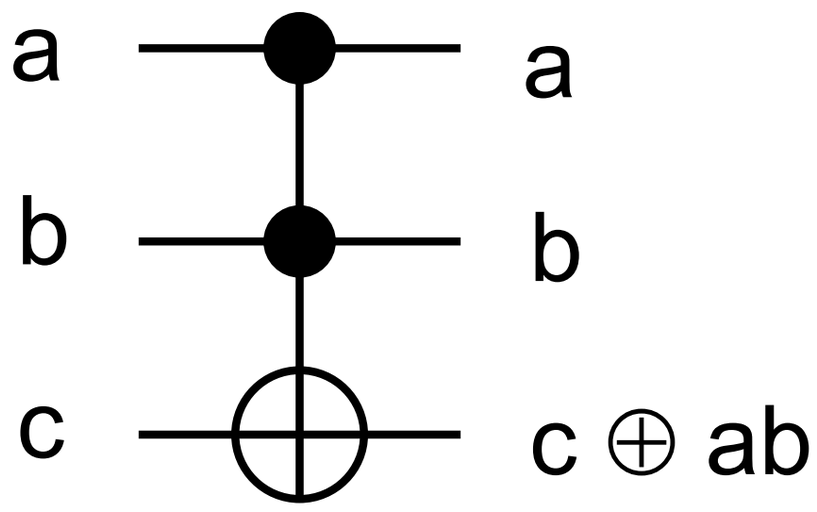


$$U_{toffoli} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

$$|000\rangle \rightarrow |000\rangle; \quad |001\rangle \rightarrow |001\rangle; \quad |010\rangle \rightarrow |010\rangle; \quad |011\rangle \rightarrow |011\rangle$$
$$|100\rangle \rightarrow |100\rangle; \quad |101\rangle \rightarrow |101\rangle; \quad |110\rangle \rightarrow |111\rangle; \quad |111\rangle \rightarrow |110\rangle$$

Toffoli Gate (CCN)

- (a,b,c) → (a,b,c ⊕ ab)
- Smallest universal reversible (classic) operation
  - Functionally complete (can mimic a NAND gate)



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

$|000\rangle \rightarrow |000\rangle;$ $|001\rangle \rightarrow |001\rangle;$ $|010\rangle \rightarrow |010\rangle;$ $|011\rangle \rightarrow |011\rangle$

$|100\rangle \rightarrow |100\rangle;$ $|101\rangle \rightarrow |101\rangle;$ $|110\rangle \rightarrow |111\rangle;$ $|111\rangle \rightarrow |110\rangle$

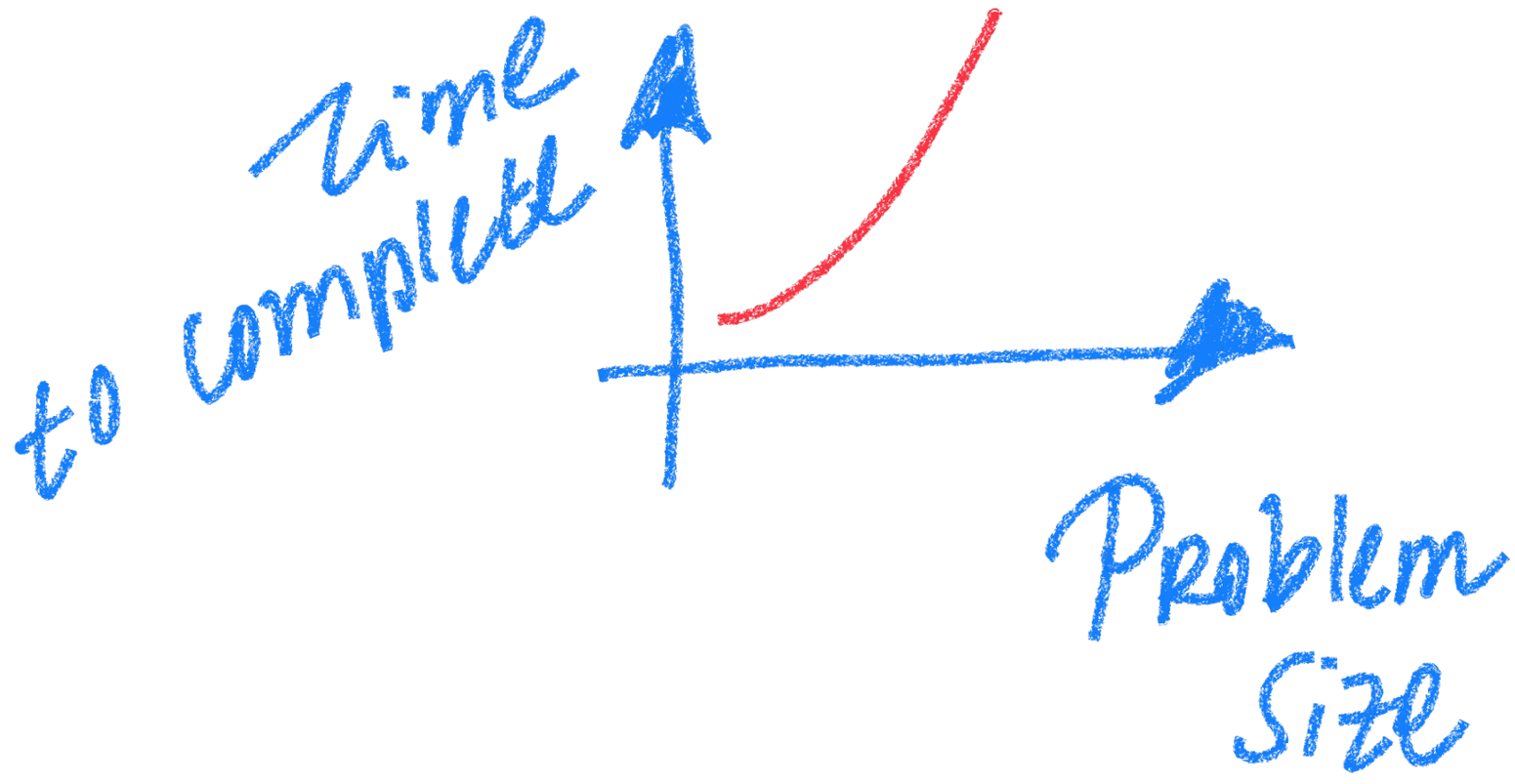
Classic Computing on Quantum Computers

- Replace all logic by an equivalent, consisting of reversible elements
  - CCN(Toffoli) gate
- Any deterministic classic computation can be mapped
- Non-deterministic classic computation?
  - I.e., random bits are generated to be used in computation
  - How to simulate the outcome of a random fair coin toss?
    - Prepare a qubit in state |0>
    - Send it through a Hadamard gate to produce

$$H|0>=\frac{1}{\sqrt{2}}|0>+\frac{1}{\sqrt{2}}|1>=\frac{1}{\sqrt{2}}(|0>+|1>)$$

- Measure the state: The result is 0 or 1 with 50/50 probability

Functionality vs. Efficiency



n qbits vs. n bits

$|\cdot\rangle$

$\begin{bmatrix} \vdots \\ \cdot \end{bmatrix}$

2^n x complex numbers

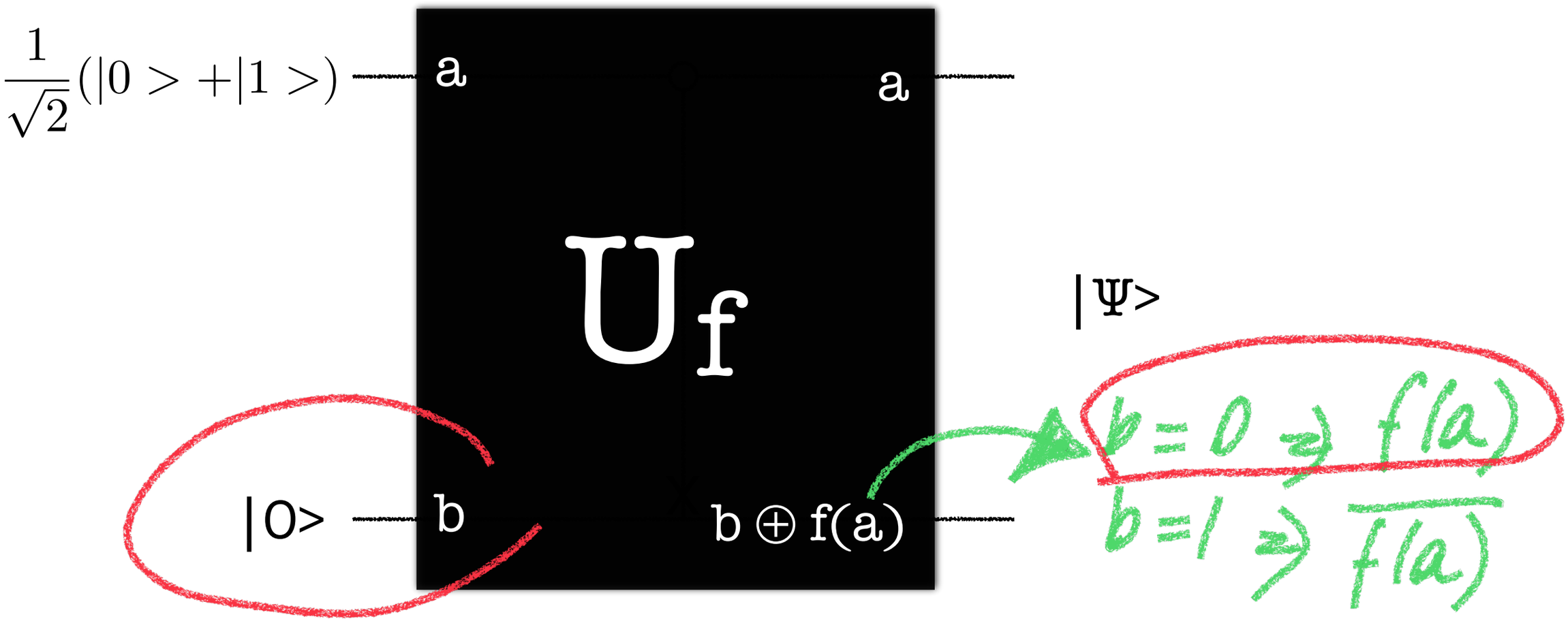
0 1 ... 1 1

n

Quantum ||ism

- f(x): {0,1} → {0,1}
- How to compute f(x) on a quantum computer?
  - Two-qubit computer, initial state: |Ψ> = |ab>
  - Sequence of unitary transformations: U_f
    - |ab> → |a, b ⊕ f(a)>
    - if b=0, second qbit state becomes |f(a)>

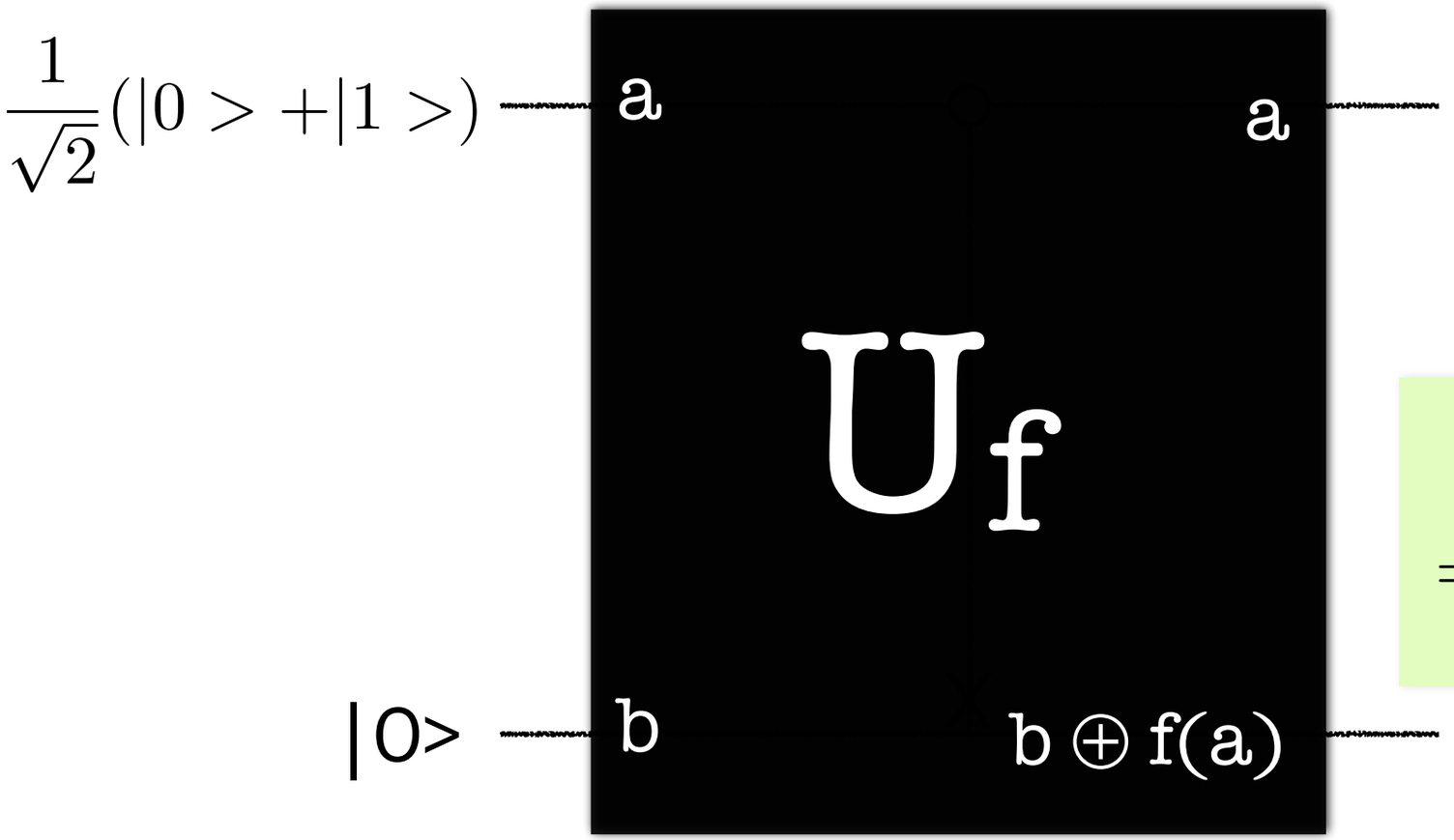
a	b	a	b ⊕ f(a)
0	0	0	f(a)=f(0)
0	1	0	not(f(a))
1	0	1	f(a)=f(1)
1	1	1	not(f(a))



Quantum ||ism

- f(x): {0,1} → {0,1}
- How to compute f(x) on a quantum computer?
  - Two-qubit computer, initial state: $|\Psi\rangle = |ab\rangle$
  - Sequence of unitary transformations: U_f
    - $|ab\rangle \rightarrow |a, b \oplus f(a)\rangle$
    - if b=0, second qbit state becomes $|f(a)\rangle$

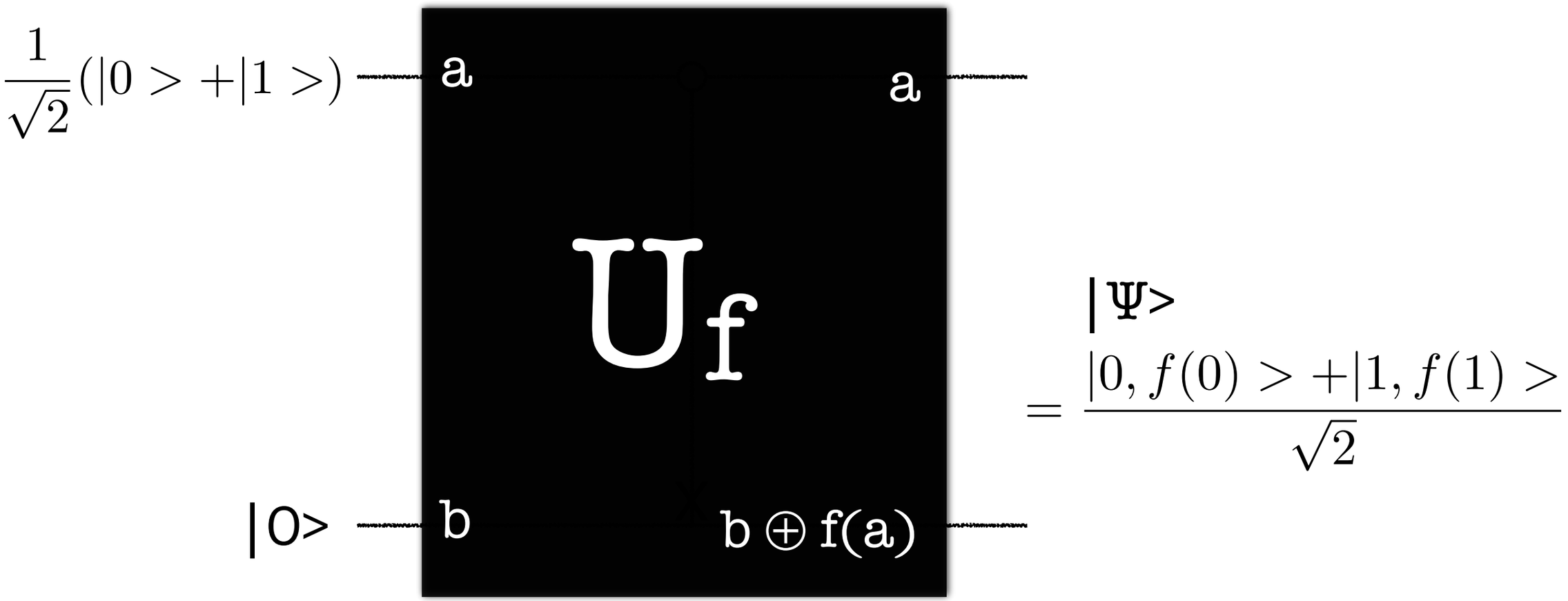
a	b	a	b ⊕ f(a)
0	0	0	f(a)=f(0)
0	1	0	not(f(a))
1	0	1	f(a)=f(1)
1	1	1	not(f(a))



$$= \frac{|\Psi\rangle}{\frac{|0, f(0)> + |1, f(1)>}{\sqrt{2}}}$$

Quantum ||ism

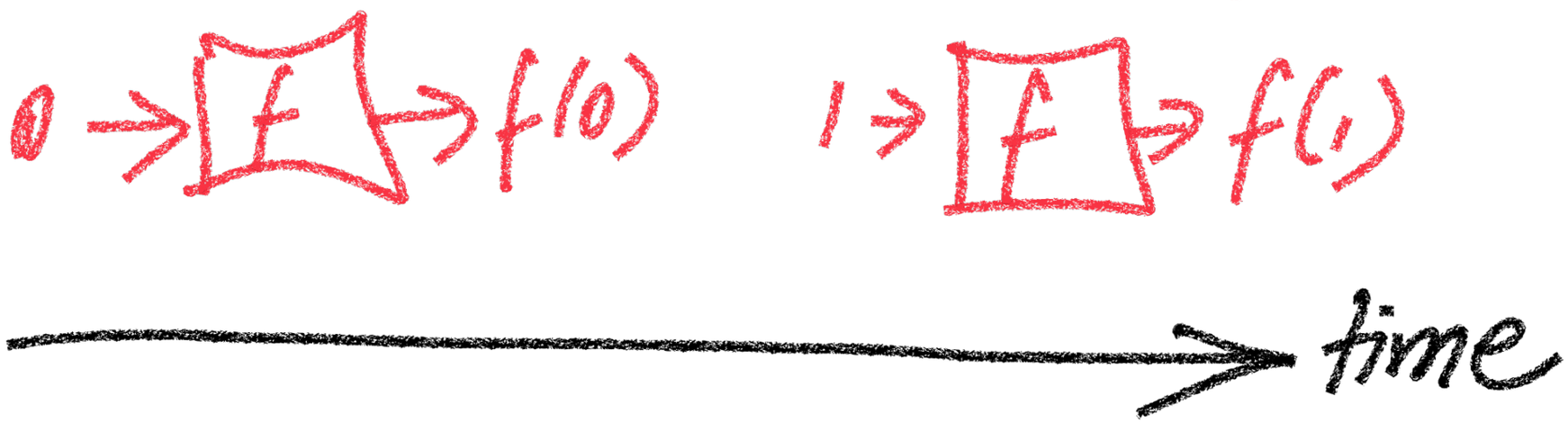
- Classic parallelism
  - **Multiple** f(x) circuits evaluate f(x) for a **single** value of x simultaneously
- Quantum parallelism
  - **Single** f(x) circuit evaluates f(x) for **multiple** values of x simultaneously

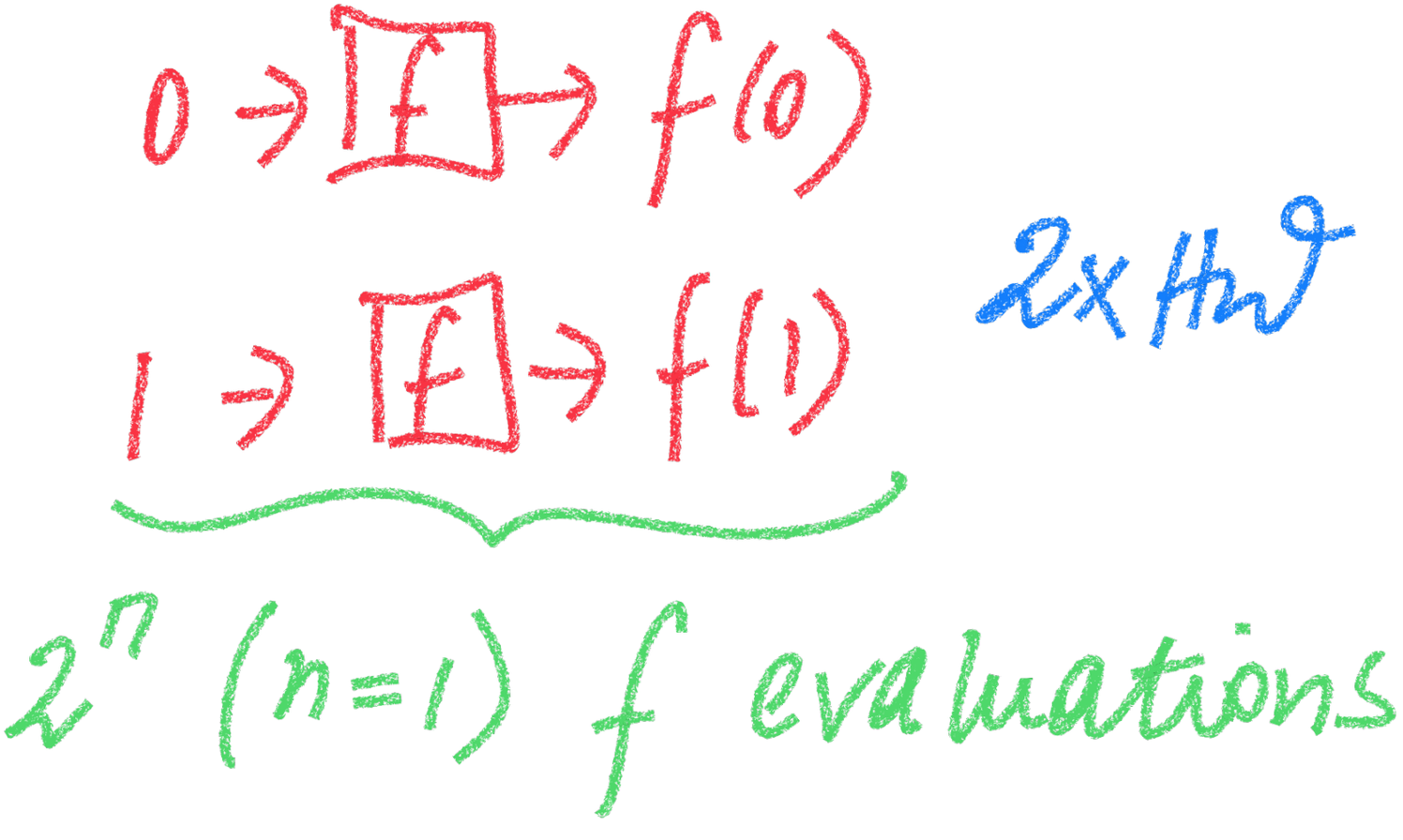


vs. classical llism

1x HW → HW implementing f

f(0)
f(1)





Measurement 2:

Only $f(0)$ or $f(1)$!

State after U_f transformation:

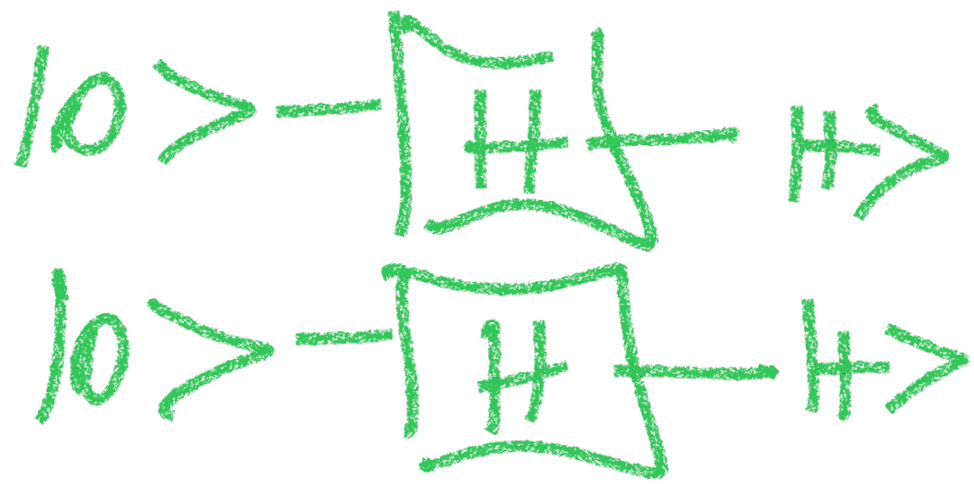
$$\frac{|0, p(0)\rangle + |1, p(1)\rangle}{\sqrt{2}}$$

Quantum ||ism: Multi-bit generalization

- Function f of arbitrary number of bits?
- Hadamard transform
  - n Hadamard gates acting simultaneously on n qubits
  - n = 2 → $H^{\otimes 2}$

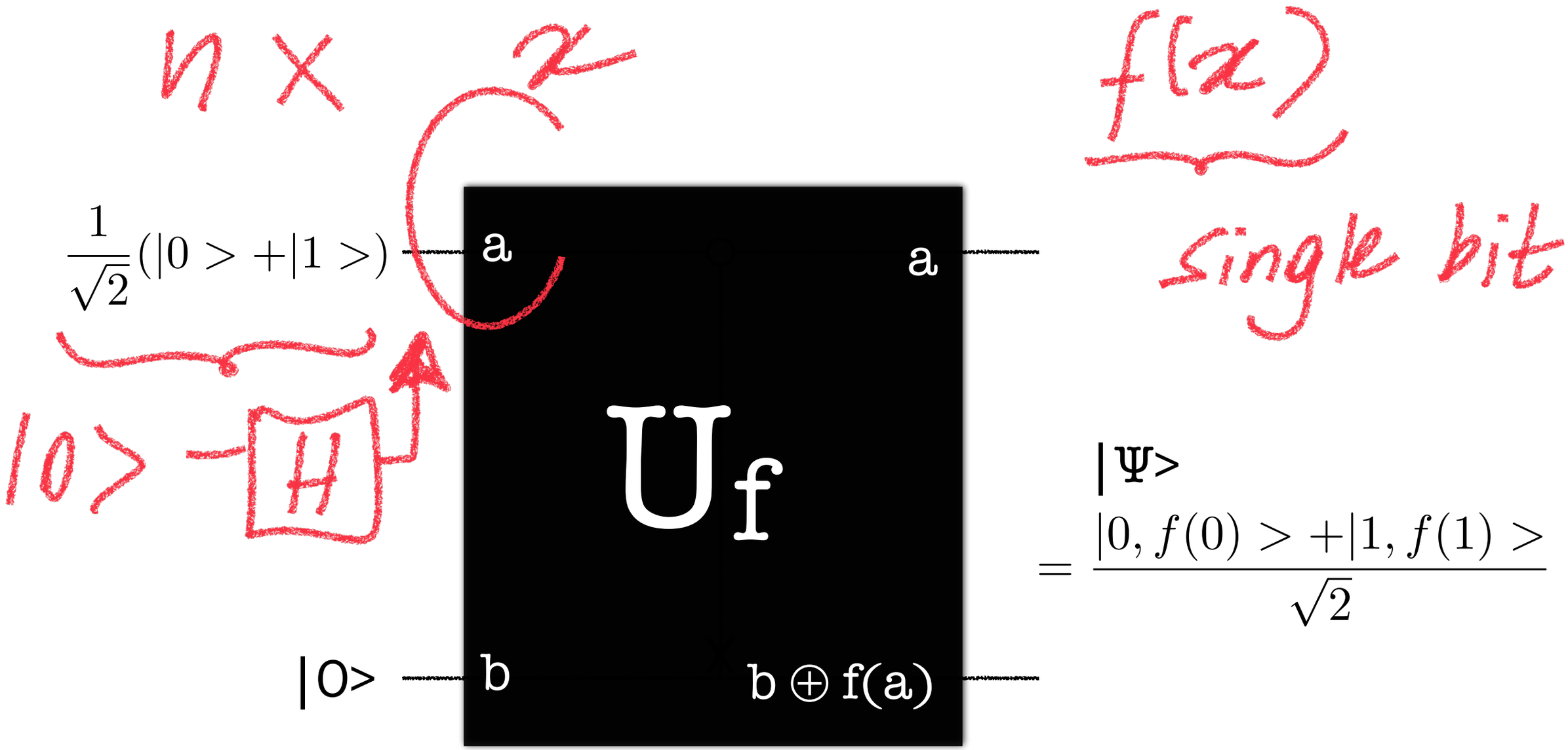
$$\frac{|0>+|1>}{\sqrt{2}}\frac{|0>+|1>}{\sqrt{2}} = \frac{|00>+|01>+|10>+|11>}{2}$$

- Hadamard transform $H^{\otimes n}$ on n qbits all initialized to |0>: $\frac{1}{\sqrt{2^n}}\sum_x |x>$
  - Sum over all possible values of x
- Equal superposition of all computational basis states
  - Superposition of 2^n states using n gates

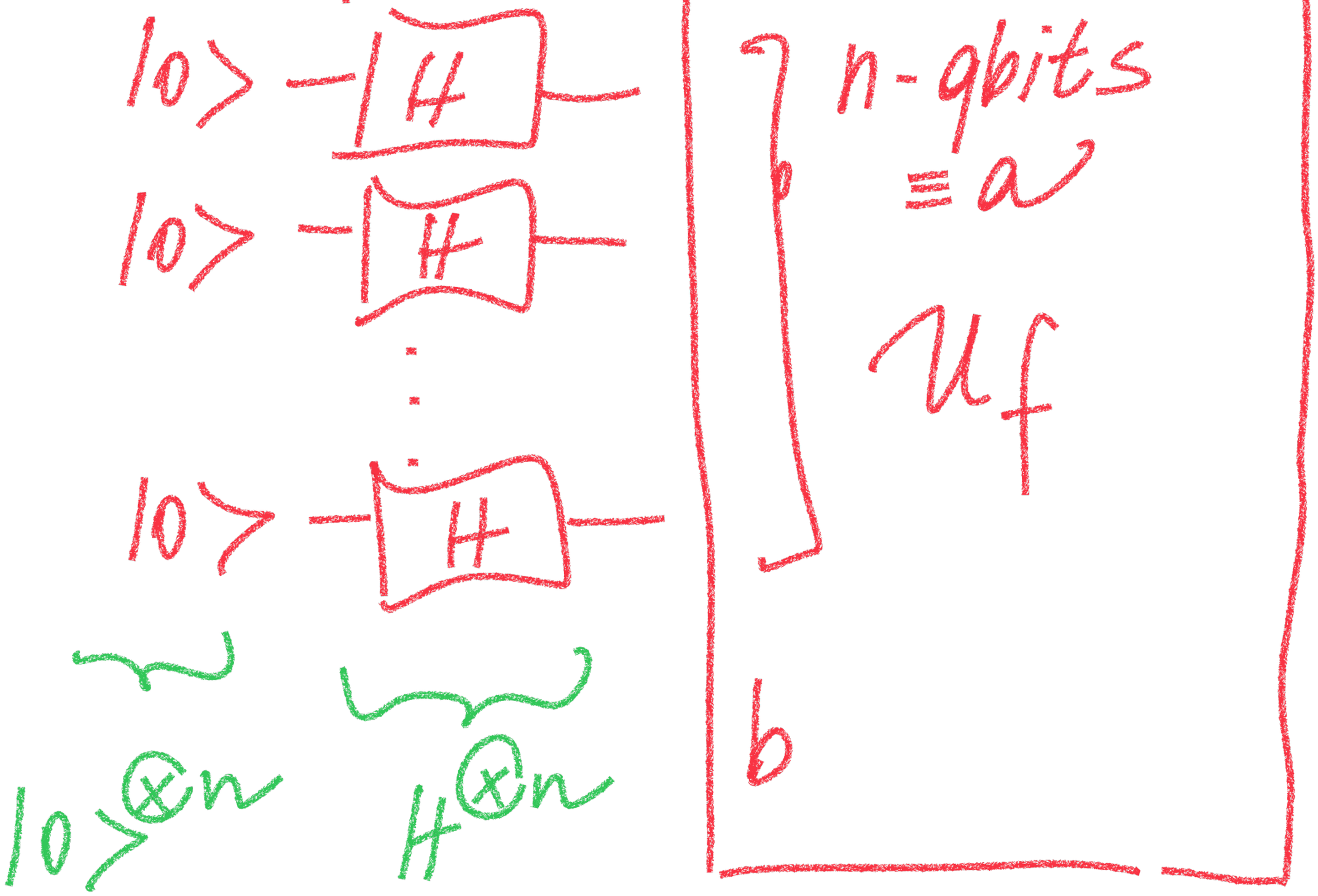


Quantum ||ism

- Classic ||ism
  - **Multiple** f(x) circuits evaluate f(x) for a **single** value of x simultaneously
- Quantum ||ism
  - **Single** f(x) circuit evaluates f(x) for **multiple** values of x simultaneously



Multi-qbit vs. classical $f(x)$ n bits

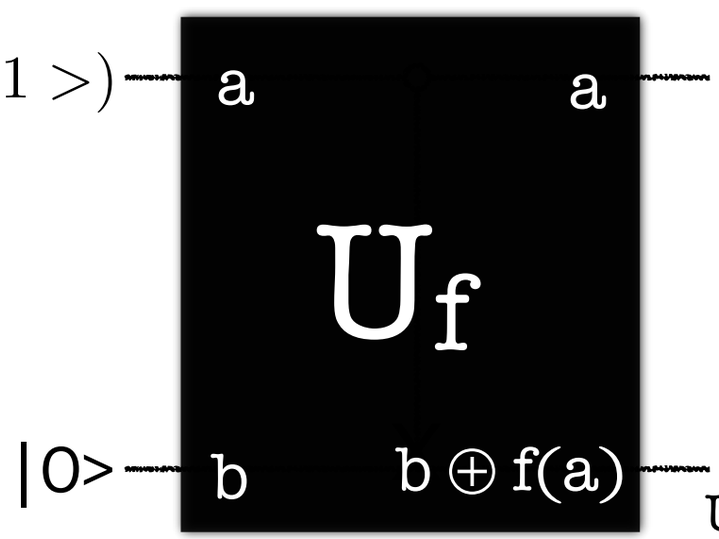


Quantum ||ism: Multi-bit generalization

- Hadamard transform $H^{\oplus n}$ on n qbits all initialized to |0>: $\frac{1}{\sqrt{2^n}} \sum_x |x>$
  - Sum over all possible values of x
  - Equal superposition of all computational basis states
    - Superposition of 2^n states using n gates

- Quantum || evaluation of f(x) with n-bit input, single-bit output:
  - Prepare the n+1 qbit state $|0>^{\otimes n} |0>$
  - Apply Hadamard transformation on the first n qbits
  - Apply (multi-qbit) U_f on the resulting n qbits
  - |Ψ> becomes $\frac{1}{\sqrt{2^n}} \sum_x |x> |f(x)>$

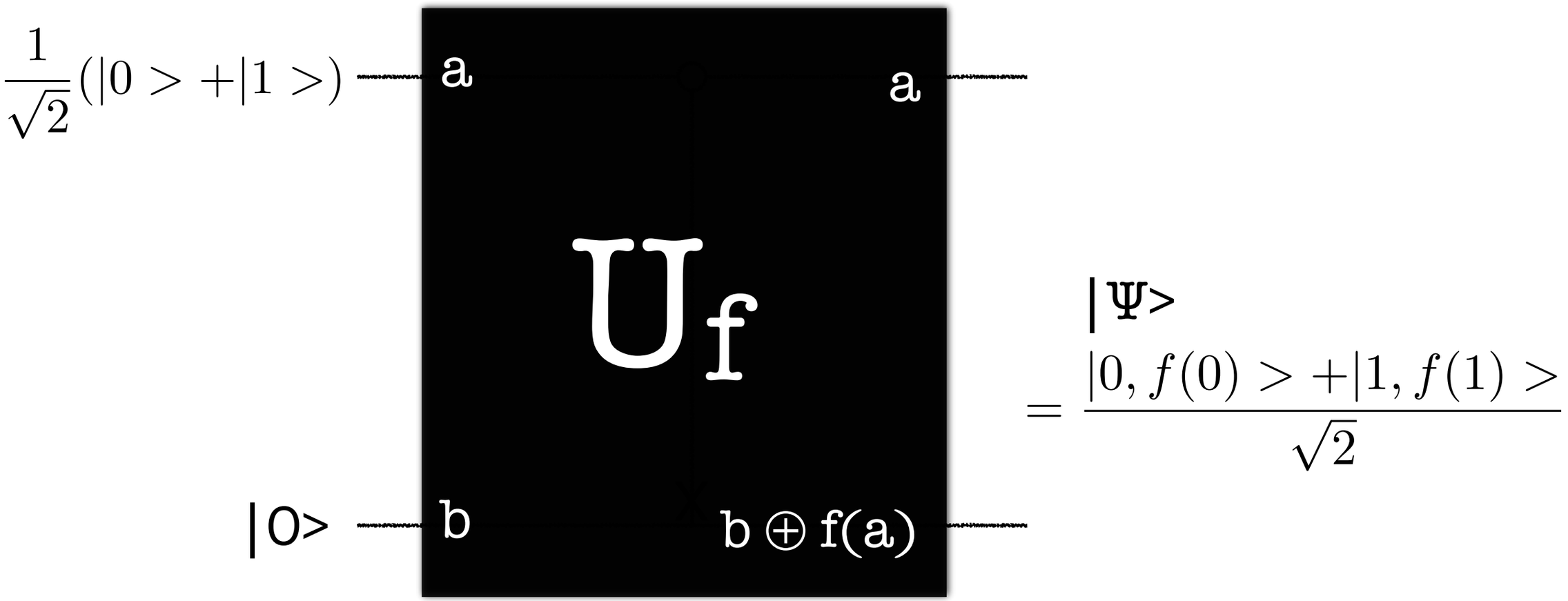
Single-bit: $\frac{1}{\sqrt{2}}(|0> + |1>)$



$|\Psi> = \frac{|0, f(0)> + |1, f(1)>}{\sqrt{2}}$

Quantum ||ism

- Classic ||ism
  - **Multiple** f(x) circuits evaluate f(x) for a **single** value of x simultaneously
- Quantum ||ism
  - **Single** f(x) circuit evaluates f(x) for **multiple** values of x simultaneously



Quantum ||ism

$$\frac{|0, f(0)> + |1, f(1)>}{\sqrt{2}}$$

Single qbit

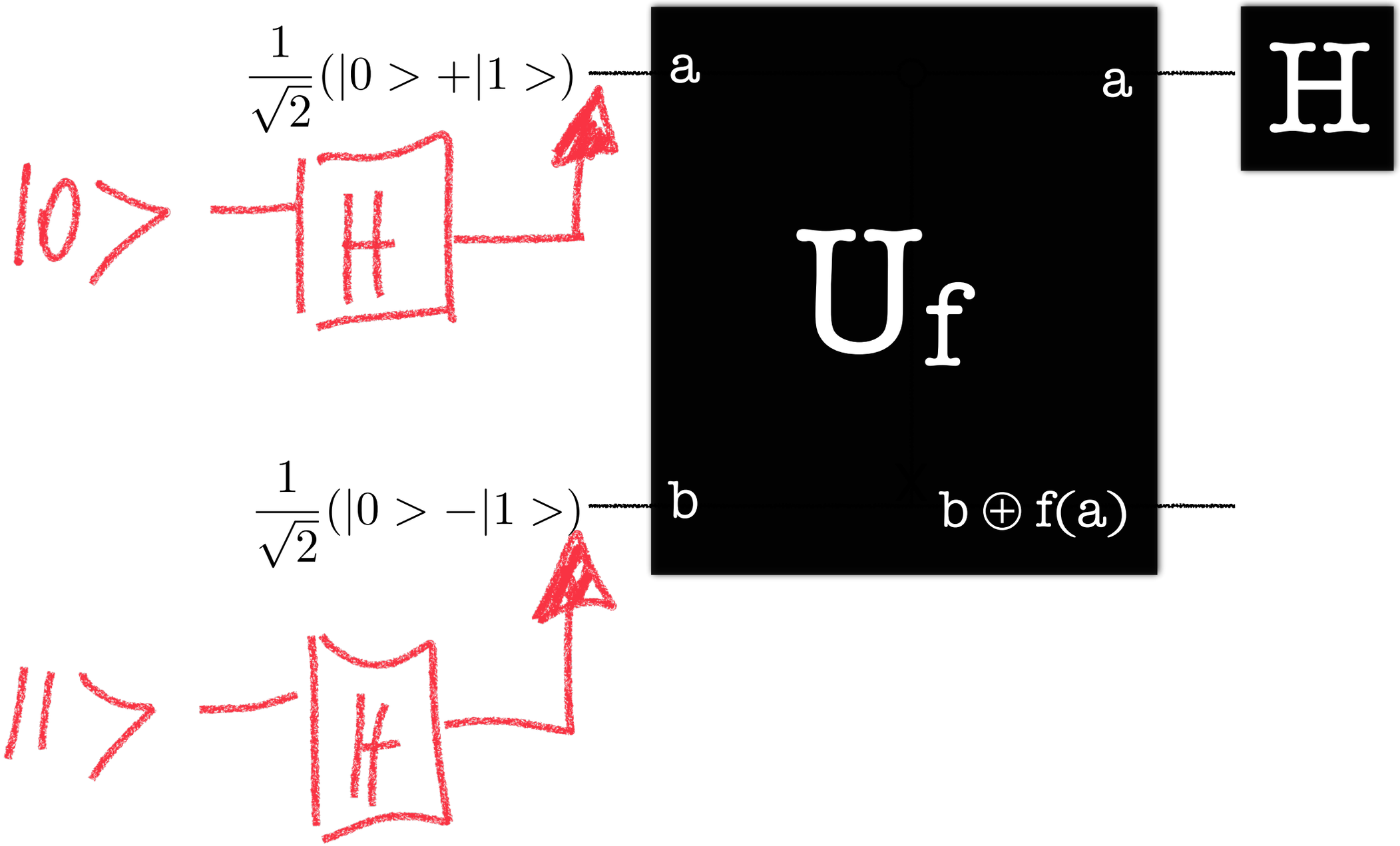
$$\frac{1}{\sqrt{2^n}} \sum_x |x> |f(x)>$$

multi qbit

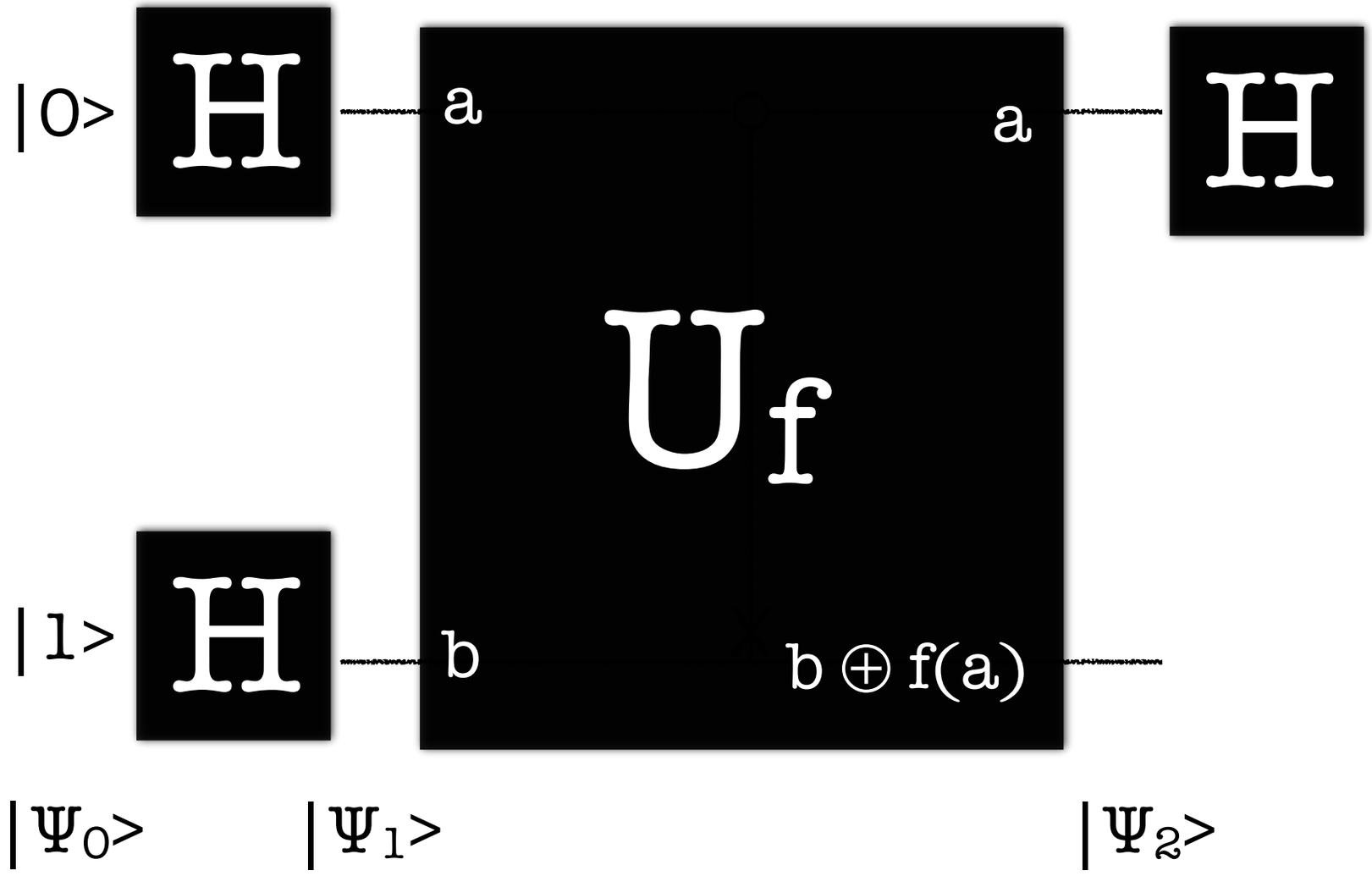
- Single qbit
  - Measurement of the output state renders either |0, f(0)> or |1, f(1)>, not both.
- Multi-qbit
  - Measurement of the output state renders a single value of f(x), not all.

Not useful unless we can extract information about more than one value of f(x) from superposition states.

Deutsch's Algorithm

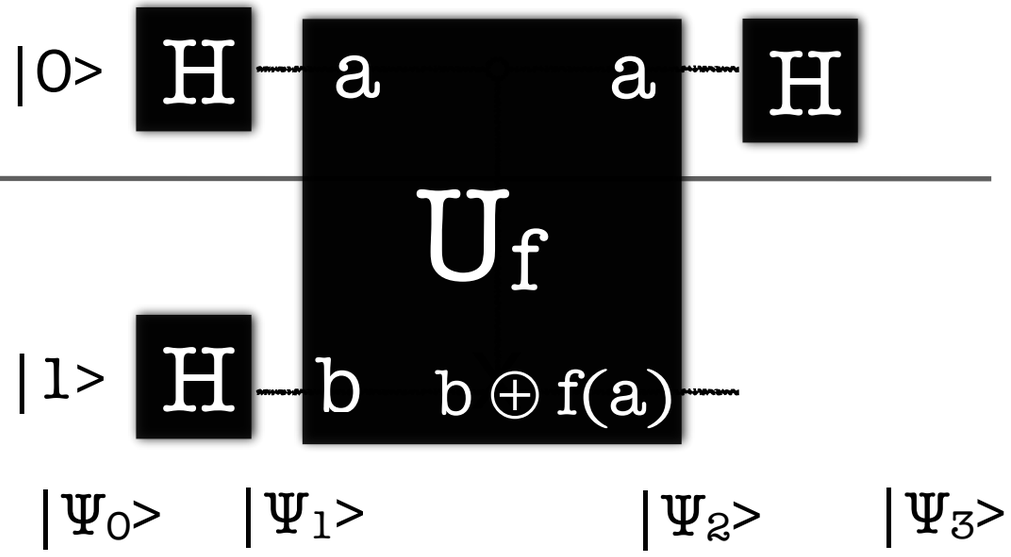


Deutsch's Algorithm



Deutsch's Algorithm

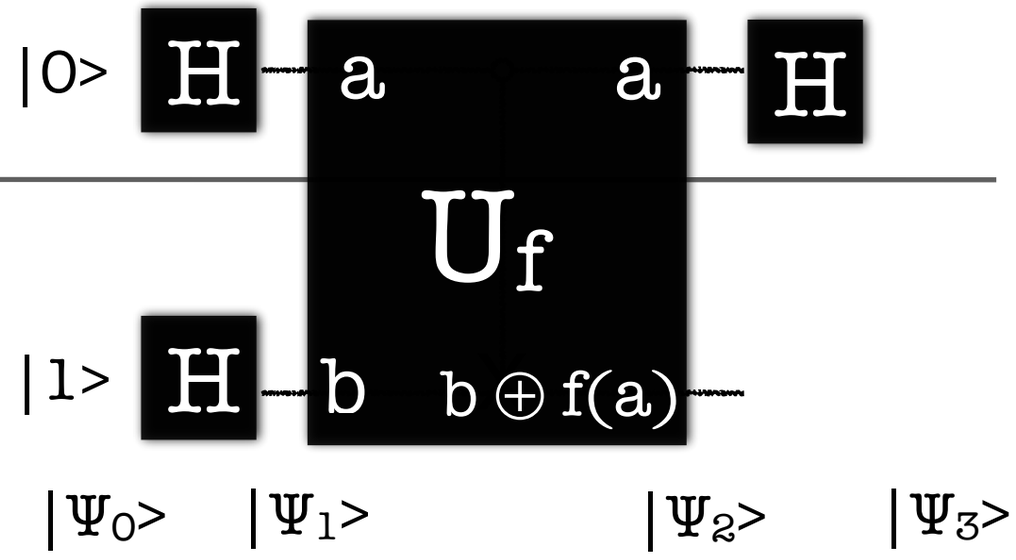
$$|\Psi_0 >= |01 >$$



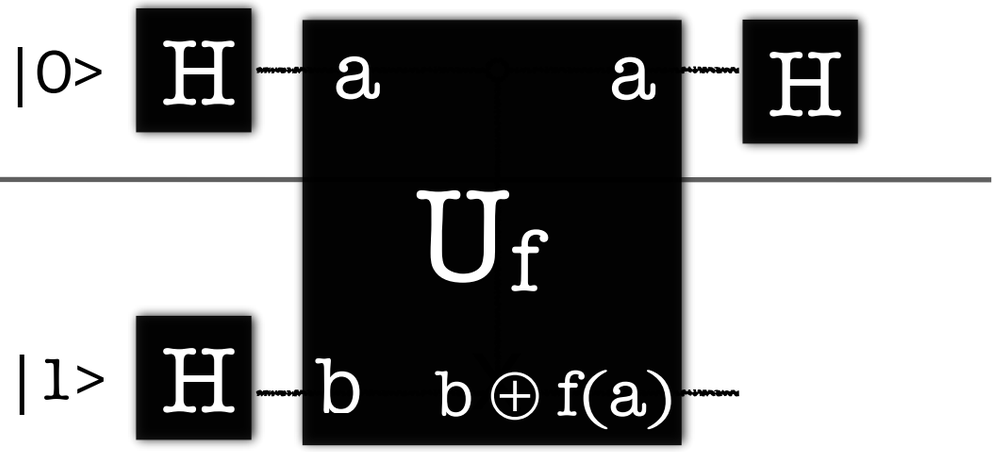
Deutsch's Algorithm

$$|\Psi_0>=|01>$$

$$|\Psi_1>=[\frac{|0>+|1>}{\sqrt{2}}][\frac{|0>-|1>}{\sqrt{2}}]$$



Deutsch's Algorithm



$$|\Psi_0>=|01>$$

$$|\Psi_1>=[\frac{|0>+|1>}{\sqrt{2}}][\frac{|0>-|1>}{\sqrt{2}}]$$

U_f on state $|a>[\frac{|0>-|1>}{\sqrt{2}}]$?

$|a>|b\oplus f(a)>$

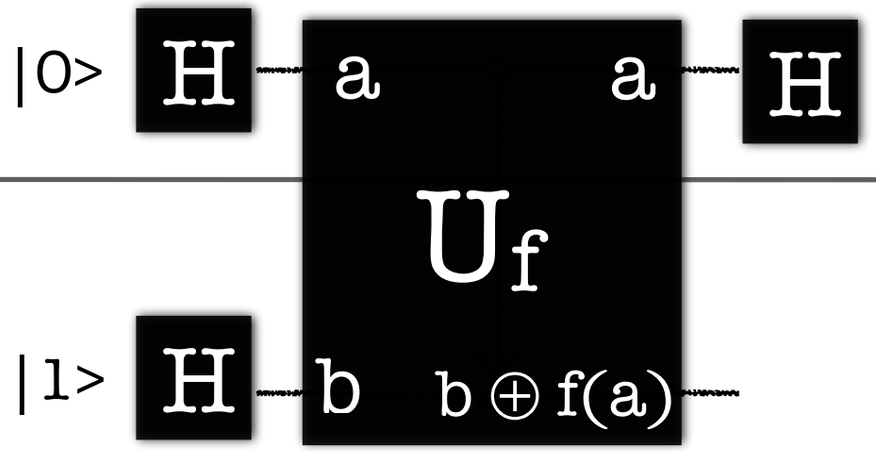
$b=0$: $f(a)$

$b=1$: $\overline{f(a)}$

$f(a)$ → Classical func.

a	b	a	b ⊕ f(a)
0	0	0	f(a)
0	1	0	not(f(a))
1	0	1	f(a)
1	1	1	not(f(a))

Deutsch's Algorithm



$$|\Psi_0>=|01>$$

$$|\Psi_1>=[\frac{|0>+|1>}{\sqrt{2}}][\frac{|0>-|1>}{\sqrt{2}}]$$

U_f on state $|a>[\frac{|0>-|1>}{\sqrt{2}}]$?

$|a>\cdot 1/\sqrt{2}\cdot[|f(a)>-|\overline{f(a)}>]$ $\begin{matrix} b=0 : f(a) \\ b=1 : \overline{f(a)} \end{matrix}$

$$|a\rangle \cdot 1/\sqrt{2}\,\underbrace{[\,|f(a)\rangle - |\overline{f(a)}\rangle\,]}$$

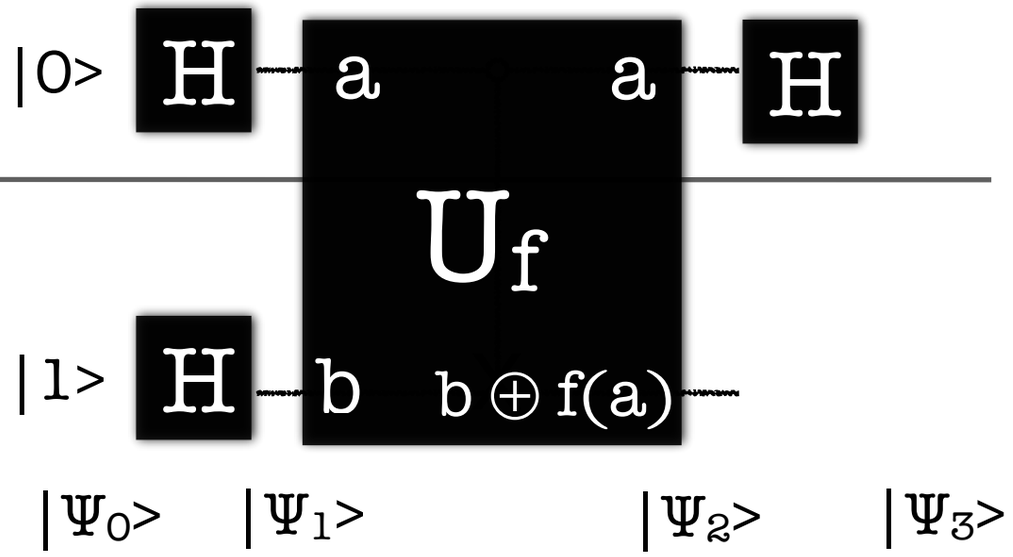
$$\left.\begin{aligned} f(a)=0 &: [\,|0\rangle - |1\rangle\,] \\ f(a)=1 &: [\,|1\rangle - |0\rangle\,] \end{aligned}\right\} (-1)^{f(a)}\,[\,|0\rangle - |1\rangle\,]$$

$$\Rightarrow \underbrace{|a\rangle \cdot 1/\sqrt{2}\;(-1)^{f(a)}\,[\,|0\rangle - |1\rangle\,]}$$

$a=0:\ |0\rangle \qquad (-1)^{f(0)}\ |\!-\!\rangle$

$a=1:\ |1\rangle \qquad (-1)^{f(1)}\ |\!-\!\rangle$

Deutsch's Algorithm



$$|\Psi_0>=|01>$$

$$|\Psi_1>=[\frac{|0>+|1>}{\sqrt{2}}][\frac{|0>-|1>}{\sqrt{2}}]$$

U_f on state $|a>[\frac{|0>-|1>}{\sqrt{2}}]$?

$$(-1)^{f(a)}|a>[\frac{|0>-|1>}{\sqrt{2}}]$$

$|a> \mapsto (-1)^{f(a)}|a>$

a	b	a	b ⊕ f(a)
0	0	0	f(a)
0	1	0	not(f(a))
1	0	1	f(a)
1	1	1	not(f(a))

$$1/\sqrt{2}\,[\,|0\rangle\,(-1)^{f(0)} + |1\rangle\,(-1)^{f(1)}\,] \mapsto$$

if $f(0) = f(1)$ $\quad = 0 \rightarrow |0\rangle + |1\rangle$

$\quad = 1 \rightarrow -[|0\rangle + |1\rangle]$

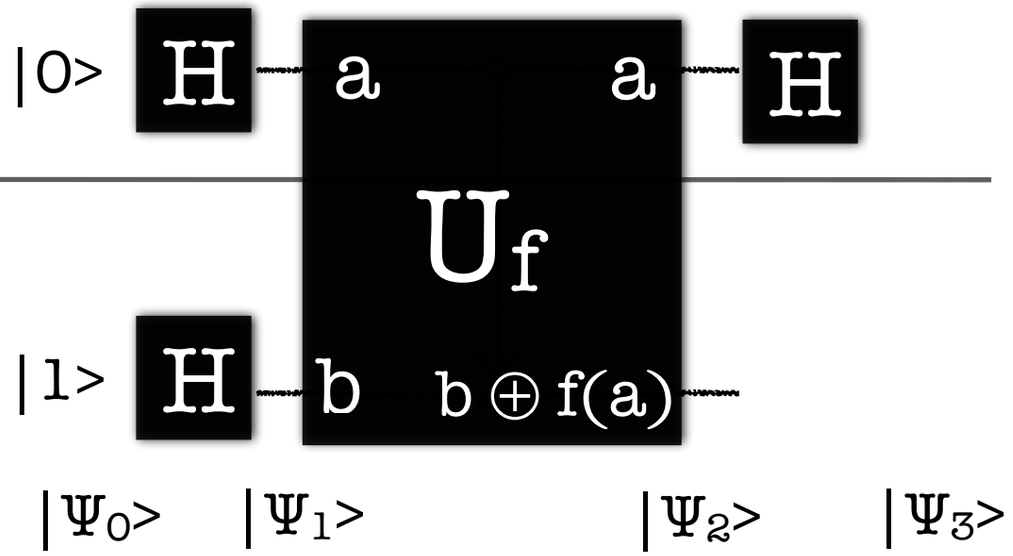
$\pm[|0\rangle + |1\rangle]$

$f(0) \neq f(1)$ $\rightarrow \pm[|0\rangle - |1\rangle]$

$\Rightarrow \pm 1/\sqrt{2}\,[|0\rangle + |1\rangle] \mapsto f(0) = f(1)$

$\pm 1/\sqrt{2}\,[|0\rangle - |1\rangle] \mapsto f(0) \neq f(1)$

Deutsch's Algorithm



$$|\Psi_0>=|01>$$

$$|\Psi_1>=[\frac{|0>+|1>}{\sqrt{2}}][\frac{|0>-|1>}{\sqrt{2}}]$$

U_f on state $|a>[\frac{|0>-|1>}{\sqrt{2}}]$? $(-1)^{f(a)}|a>[\frac{|0>-|1>}{\sqrt{2}}]$

$$|\Psi_2>=\begin{cases}\pm[\frac{|0>+|1>}{\sqrt{2}}][\frac{|0>-|1>}{\sqrt{2}}] & if \quad f(0)=f(1)\\ \pm[\frac{|0>-|1>}{\sqrt{2}}][\frac{|0>-|1>}{\sqrt{2}}] & if \quad f(0)\neq f(1)\end{cases}$$

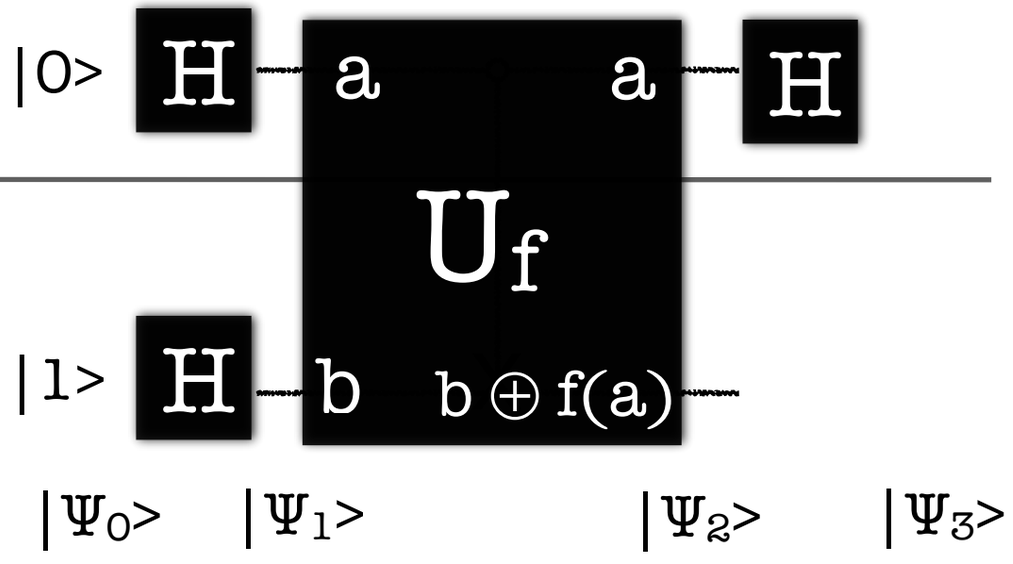
$|->$

$\pm|0> \quad f(0)=f(1)$

$\pm|1> \quad f(0)\neq f(1)$

H

Deutsch's Algorithm



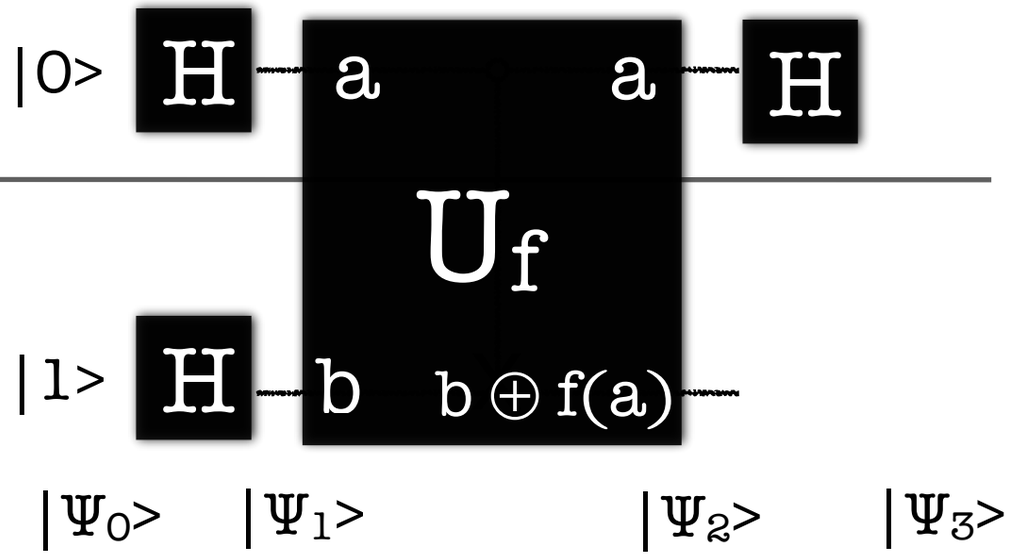
$$|\Psi_0>=|01>$$

$$|\Psi_1>=[\frac{|0>+|1>}{\sqrt{2}}][\frac{|0>-|1>}{\sqrt{2}}]$$

$$|\Psi_2>=\begin{cases}\pm[\frac{|0>+|1>}{\sqrt{2}}][\frac{|0>-|1>}{\sqrt{2}}] & if \quad f(0)=f(1)\\ \pm[\frac{|0>-|1>}{\sqrt{2}}][\frac{|0>-|1>}{\sqrt{2}}] & if \quad f(0)\neq f(1)\end{cases}$$

$$|\Psi_3>=\begin{cases}\pm|0>[\frac{|0>-|1>}{\sqrt{2}}] & if \quad f(0)=f(1)\\ \pm|1>[\frac{|0>-|1>}{\sqrt{2}}] & if \quad f(0)\neq f(1)\end{cases}$$

Deutsch's Algorithm



$$|\Psi_0>=|01>$$

$$|\Psi_1>=[\frac{|0>+|1>}{\sqrt{2}}][\frac{|0>-|1>}{\sqrt{2}}]$$

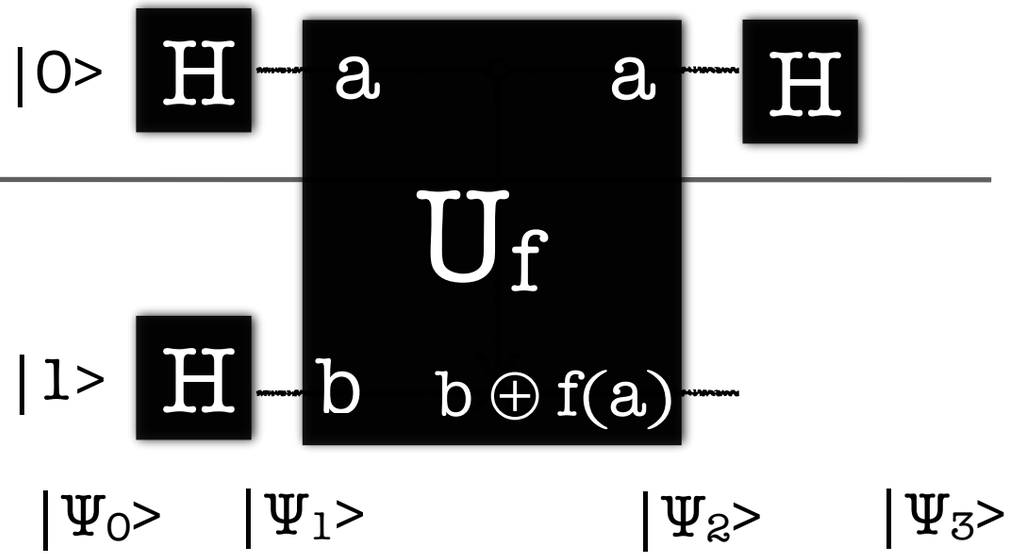
$$|\Psi_2>=\begin{cases}\pm[\frac{|0>+|1>}{\sqrt{2}}][\frac{|0>-|1>}{\sqrt{2}}] & if \quad f(0)=f(1)\\ \pm[\frac{|0>-|1>}{\sqrt{2}}][\frac{|0>-|1>}{\sqrt{2}}] & if \quad f(0)\neq f(1)\end{cases}$$

$$|\Psi_3>=\begin{cases}\pm|0>[\frac{|0>-|1>}{\sqrt{2}}] & if \quad f(0)=f(1)\\ \pm|1>[\frac{|0>-|1>}{\sqrt{2}}] & if \quad f(0)\neq f(1)\end{cases}$$

$$|\Psi_3>=\pm|f(0)\oplus f(1)>[\frac{|0>-|1>}{\sqrt{2}}]$$

Deutsch's Algorithm

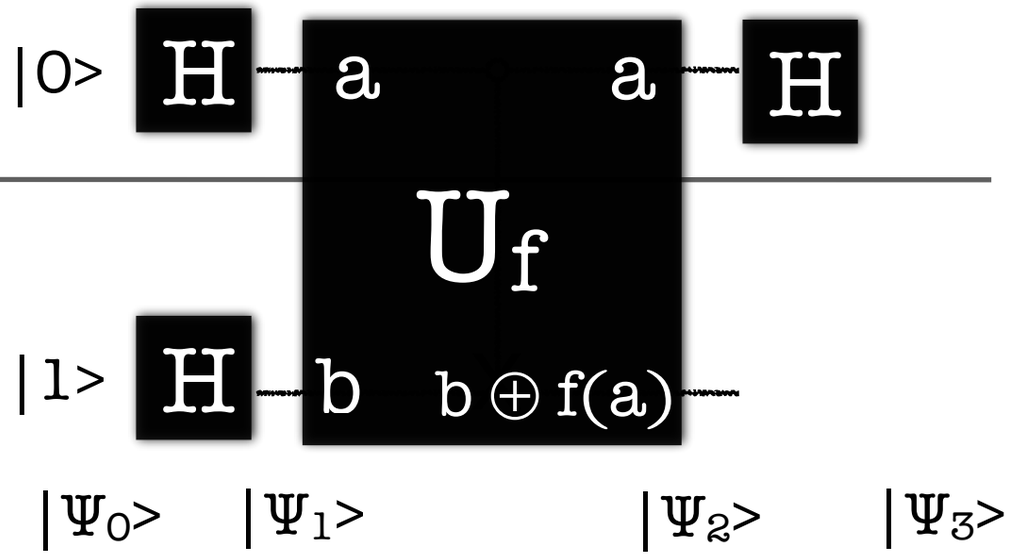
$$|\Psi_3 >= \pm|f(0) \oplus f(1) > [\frac{|0> - |1>}{\sqrt{2}}]$$



- By measuring the first qubit, we determine $f(0) \oplus f(1)$
  - A global property of f(x), using only one evaluation of f(x)
  - Classic equivalent requires at least 2 evaluations

Deutsch's Algorithm

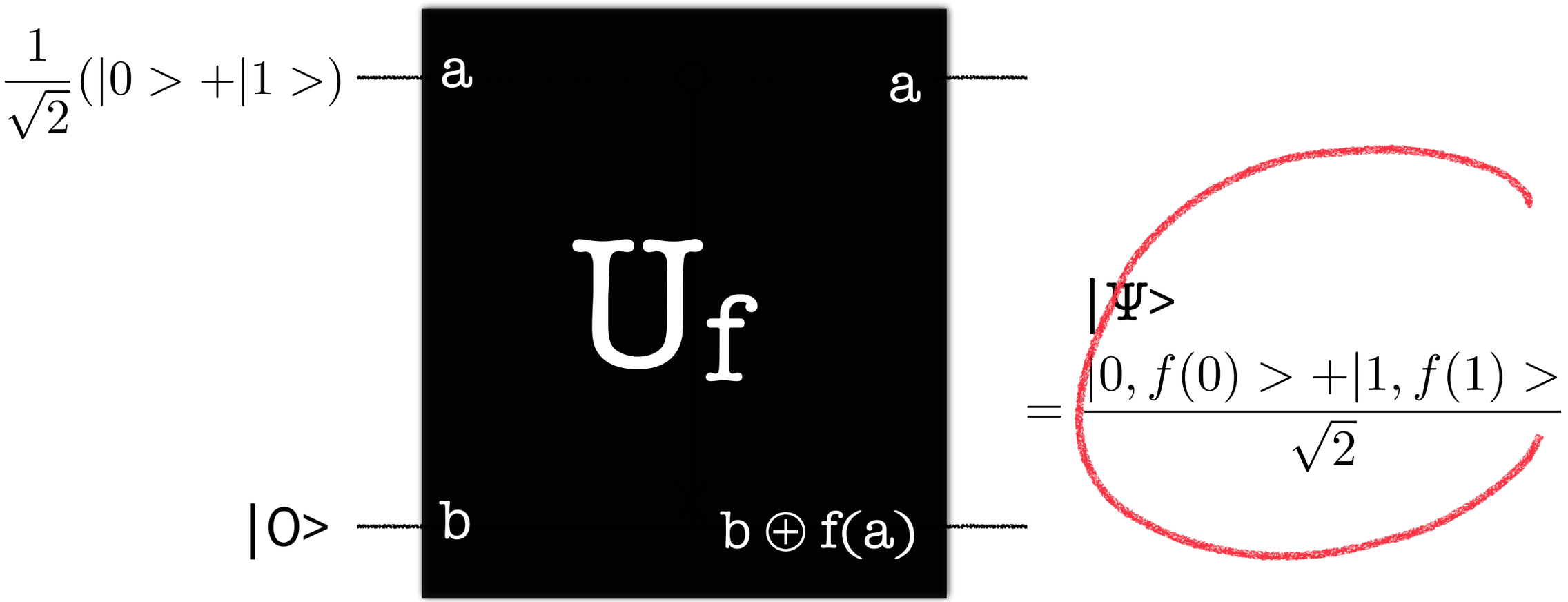
$$|\Psi_3> = \pm|f(0) \oplus f(1)> [\frac{|0> - |1>}{\sqrt{2}}]$$



- By measuring the first qubit, we determine $f(0) \oplus f(1)$
  - A global property of f(x), using only one evaluation of f(x)
  - Classic equivalent requires at least 2 evaluations

Quantum ||ism

- Classic ||ism
  - **Multiple** f(x) circuits evaluate f(x) for a **single** value of x simultaneously
- Quantum ||ism
  - **Single** f(x) circuit evaluates f(x) for **multiple** values of x simultaneously

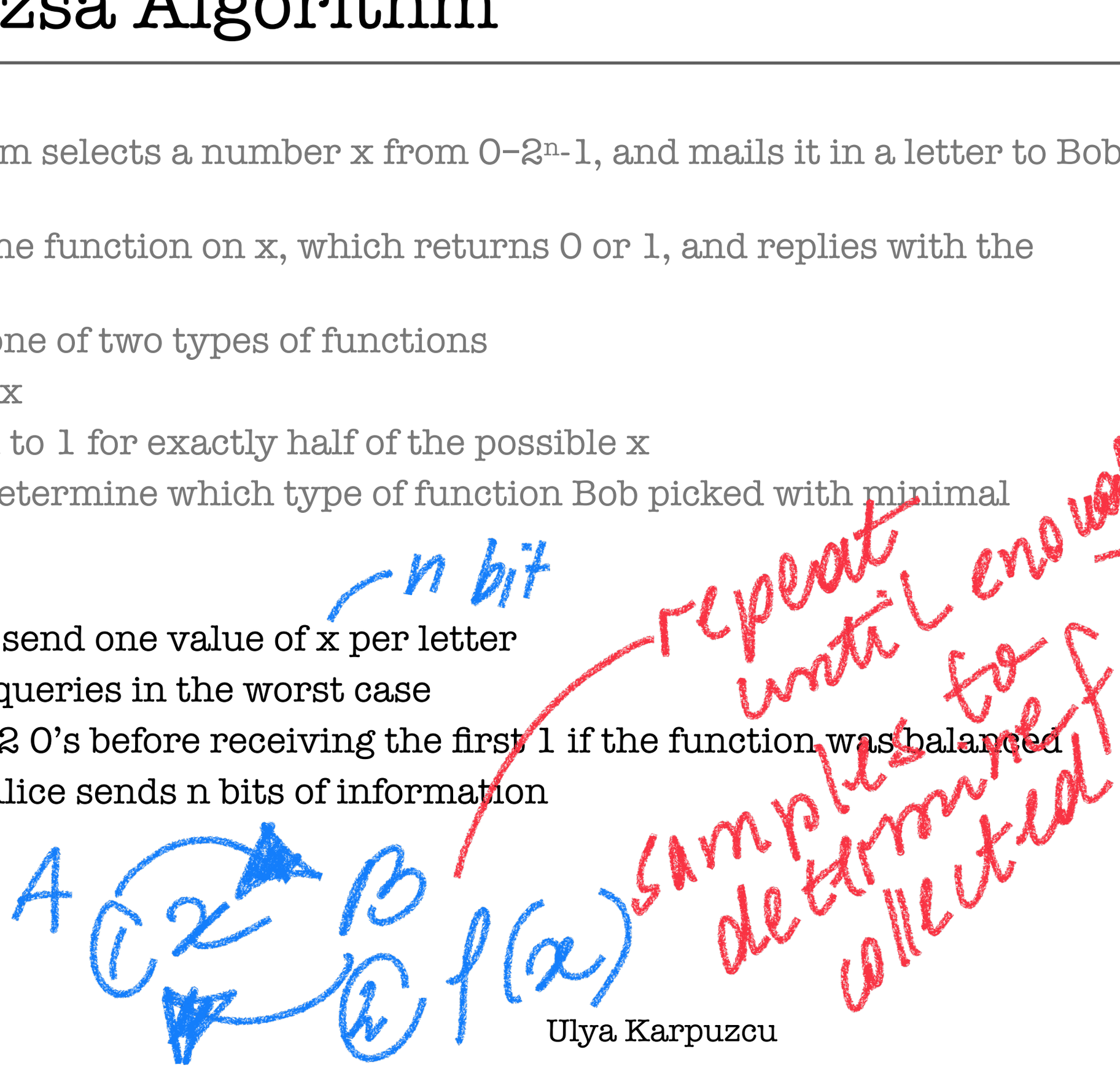


Deutsch-Jozsa Algorithm

- Problem statement:
  - A(lice) in Amsterdam selects a number x from 0–2^n-1, and mails it in a letter to B(ob) in Boston
  - B calculates some function on x, which returns 0 or 1, and replies with the result
  - B can only use one of two types of functions
    - Constant for all x
    - Balanced: Equal to 1 for exactly half of the possible x
  - A's task is to determine which type of function B picked with minimal communication

Deutsch-Jozsa Algorithm

- Problem statement:
  - Alice in Amsterdam selects a number x from 0–2^n-1, and mails it in a letter to Bob in Boston
  - Bob calculates some function on x, which returns 0 or 1, and replies with the result
  - Bob can only use one of two types of functions
    - Constant for all x
    - Balanced: Equal to 1 for exactly half of the possible x
  - Alice's task is to determine which type of function Bob picked with minimal communication
- Classic case
  - Alice may need to send one value of x per letter
    - At least $2^n/2+1$ queries in the worst case
    - May receive $2^n/2$ 0's before receiving the first 1 if the function was balanced
    - In each query, Alice sends n bits of information



Classical Solution:

Best Case: 2 samples x_1, x_2 w/ $f(x_1) \neq f(x_2)$

$\Rightarrow f$ is balanced

A receives 01 OR 10

Worst Case: $\underbrace{2^{n}/2 + 1}_{2^{n-1}+1}$ samples

$\Rightarrow f$ is balanced or const.

A receives all 0s then 1 OR 1s 0

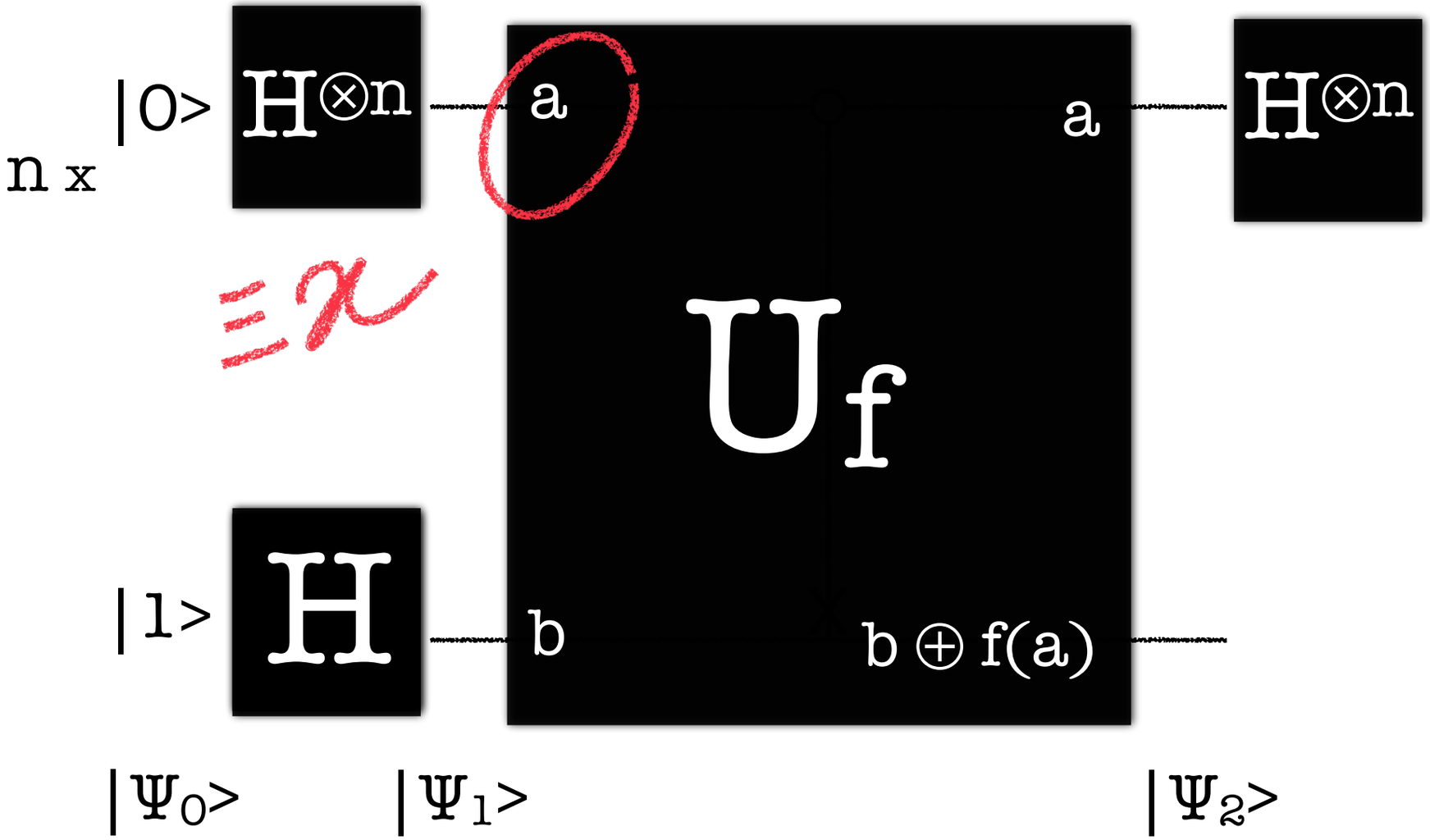
Still exponential complexity

Deutsch-Jozsa Algorithm

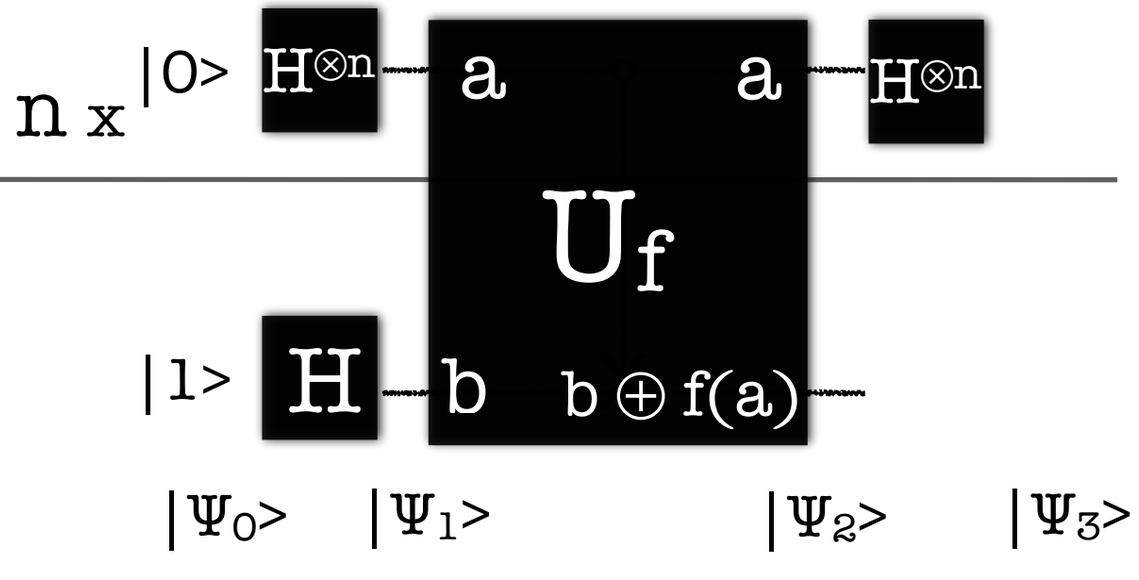
- Problem statement:
  - Alice in Amsterdam selects a number x from 0–2^n-1, and mails it in a letter to Bob in Boston
  - Bob calculates some function on x, which returns 0 or 1, and replies with the result
  - Bob can only use one of two types of functions
    - Constant for all x
    - Balanced: Equal to 1 for exactly half of the possible x
  - Alice's task is to determine which type of function Bob picked with minimal communication
- Classic case
  - Alice may need to send one value of x per letter
    - At least $2^n/2+1$ queries in the worst case
    - May receive $2^n/2$ 0's before receiving the first 1 if the function was balanced
    - In each query, Alice sends n bits of information
- Quantum case: one query may suffice
  - Alice and Bob can exchange quantum bits
  - Bob agrees to calculate the function using a unitary transformation

Deutsch-Jozsa Algorithm

- Alice has an n qubit query register and a single qubit answer register
- Alice prepares the register values in a superposition state
- Bob evaluates the function using quantum parallelism, and stores the result in the answer register
- Alice interferes the states (of the query register) in the super-position using Hadamard transform, and performs a suitable measurement to determine the type of the function



Deutsch-Jozsa



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

$$|\Psi_1\rangle = \sum_{x \in \{0,1\}^n} \frac{|x>}{\sqrt{2^n}} \left[\frac{|0> - |1>}{\sqrt{2}} \right]$$

Query register keeps the superposition of all possible values

Answer register keeps the 50-50(%) superposition of 0 and 1

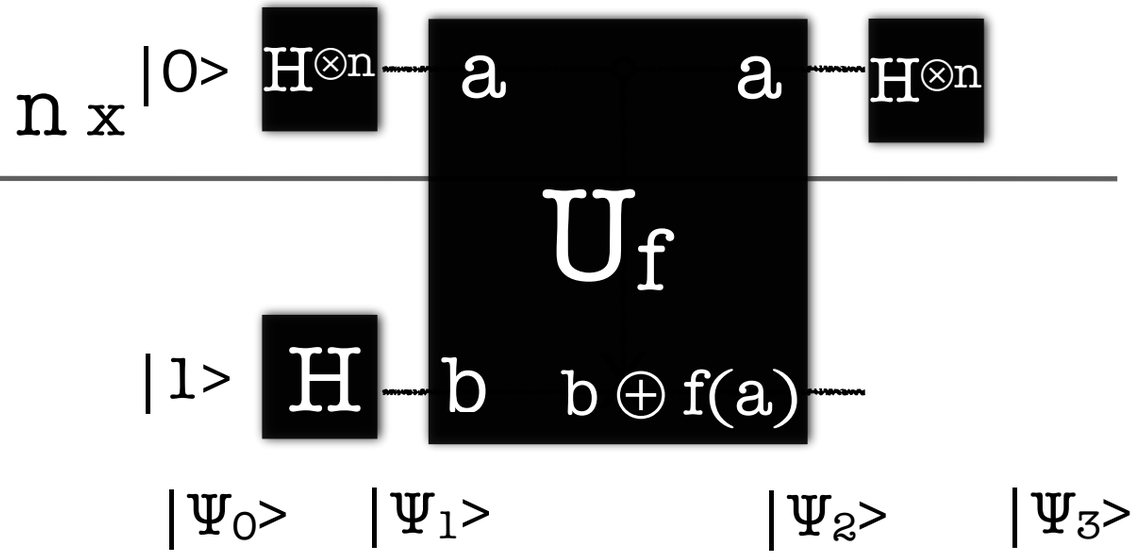
Cheat-Sheet

- Hadamard transform
  - n Hadamard gates acting simultaneously on n qubits
  - $n = 2 \rightarrow H^{\otimes 2}$

$$\frac{|0> + |1>}{\sqrt{2}} \frac{|0> + |1>}{\sqrt{2}} = \frac{|00> + |01> + |10> + |11>}{2}$$

- Hadamard transform $H^{\otimes n}$ on n qubits all initialized to |0>: $\frac{1}{\sqrt{2^n}} \sum_x |x>$
  - Sum over all possible values of x
  - Equal superposition of all computational basis states
    - Superposition of 2^n states using n gates

Deutsch-Jozsa



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

$$|\Psi_1\rangle = \sum_{x\in\{0,1\}^n} \frac{|x\rangle}{\sqrt{2^n}} \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]$$

Query register keeps the superposition of all possible values

Answer register keeps the 50-50(%) superposition of 0 and 1

Bob evaluates the function next:

$$|\Psi_2\rangle = \sum_{x} \frac{(-1)^{f(x)}|x\rangle}{\sqrt{2^n}} \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]$$

U_f on state $|a\rangle \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]$?

$(-1)^{f(a)}|a\rangle \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]$

$n>1$

$|a\rangle = \sum_x \frac{|x\rangle}{\sqrt{2^n}}$

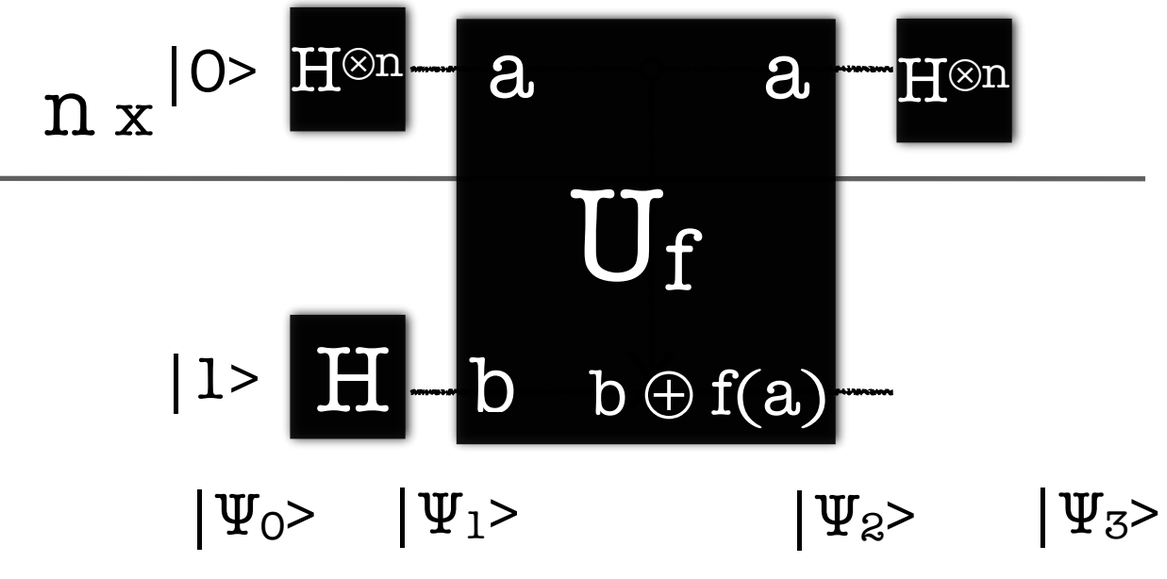
$n=1 : |a\rangle = |+\rangle$

$$(-1)^{f(a)}|a> [\frac{|0> - |1>}{\sqrt{2}}]$$

for $|a> = \sum_x \frac{|x>}{\sqrt{2^n}}$

$$\sum_x (-1)^{f(x)} \frac{|x>}{\sqrt{2^n}} \frac{|0> - |1>}{\sqrt{2}}$$

Deutsch-Jozsa



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

$$|\Psi_1\rangle = \sum_{x \in \{0,1\}^n} \frac{|x>}{\sqrt{2^n}} \left[\frac{|0> - |1>}{\sqrt{2}}\right]$$

Query register keeps the superposition of all possible values

Answer register keeps the 50-50(%) superposition of 0 and 1

Bob evaluates the function next:

$$|\Psi_2\rangle = \sum_x \frac{(-1)^{f(x)}|x>}{\sqrt{2^n}} \left[\frac{|0> - |1>}{\sqrt{2}}\right]$$

Alice applies H transform on the query register

→ $H^{\otimes n}$ → ?

Cheat-Sheet

$$H|x>= \sum_{z\in\{0,1\}} \frac{(-1)^{xz}|z>}{\sqrt{2}}$$

$n=1$:

$$\frac{(-1)^{x\cdot 0}|0>}{\sqrt{2}} + \frac{(-1)^{x\cdot 1}|1>}{\sqrt{2}}$$

$z=0$ $z=1$

$n>1$:

$$H^{\otimes n}|x_1...x_n>= H|x>= \sum_{z_1...z_n} \frac{(-1)^{x_1z_1+...+x_nz_n}|z_1...z_n>}{\sqrt{2^n}}$$

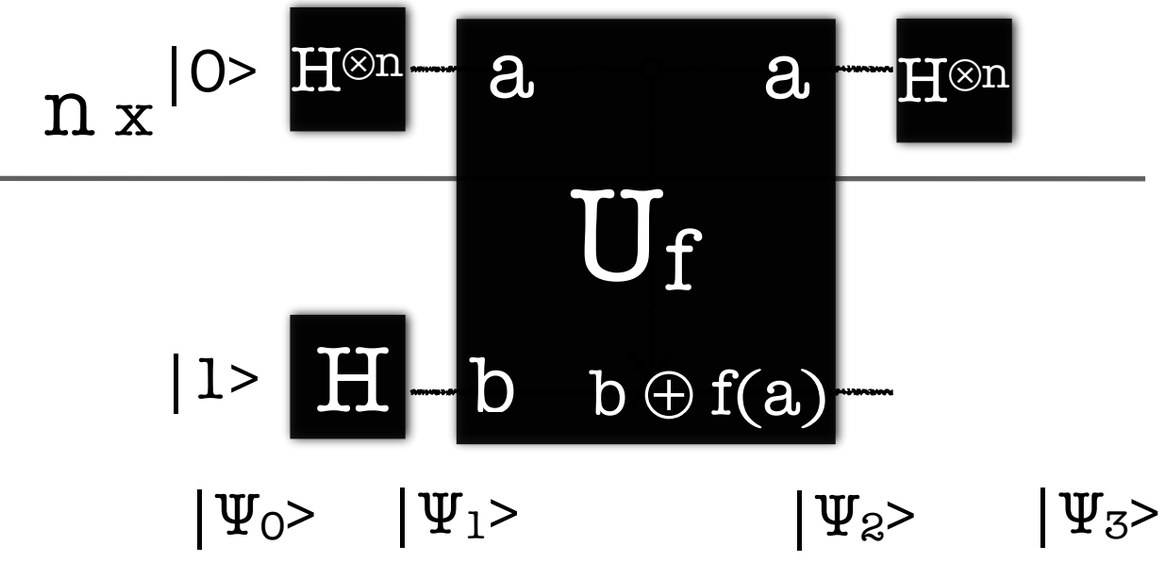
dot product

$$H^{\otimes n}|x>= H|x>= \sum_{z} \frac{(-1)^{x\bullet z}|z_1...z_n>}{\sqrt{2^n}}$$

$x=0$: $\frac{1}{\sqrt{2}}[|0>+|1>]$

$x=1$: $\frac{1}{\sqrt{2}}[|0>-|1>]$

Deutsch-Jozsa



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

$$|\Psi_1\rangle = \sum_{x\in\{0,1\}^n} \frac{|x\rangle}{\sqrt{2^n}}\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]$$

Query register keeps the superposition of all possible values

Answer register keeps the 50-50(%) superposition of 0 and 1

Bob evaluates the function next:

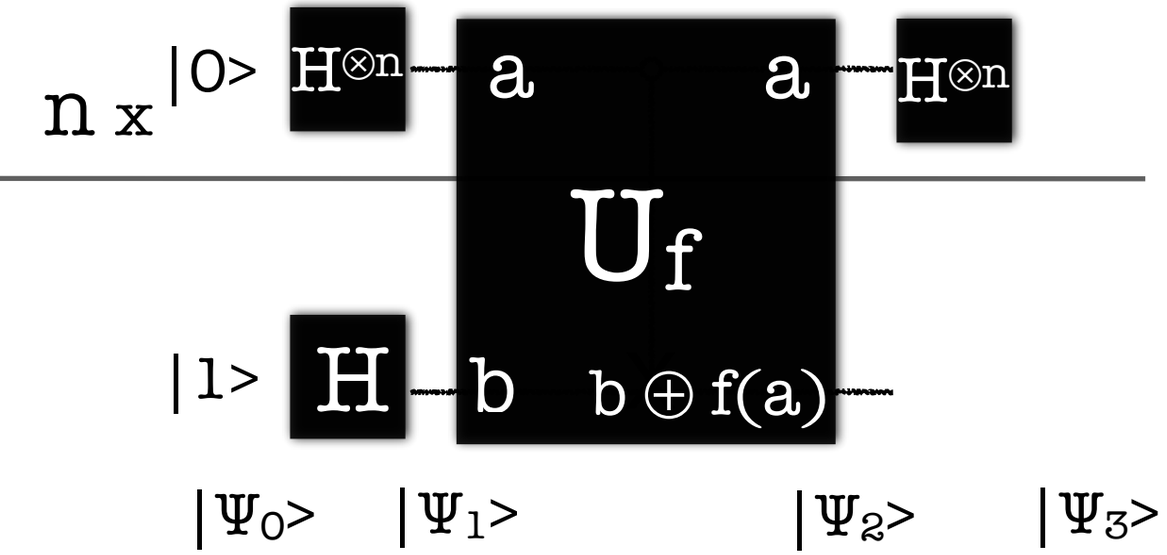
$$|\Psi_2\rangle = \sum_x \frac{(-1)^{f(x)}|x\rangle}{\sqrt{2^n}}\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]$$

$$H^{\otimes n}|x\rangle = H|x\rangle = \sum_z \frac{(-1)^{x\bullet z}|z_1 \ldots z_n\rangle}{\sqrt{2^n}}$$

Alice applies H transform on the query register

$$|\Psi_3\rangle = \sum_z \sum_x \frac{(-1)^{x\bullet z + f(x)}|z\rangle}{2^n}\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]$$

Deutsch-Jozsa

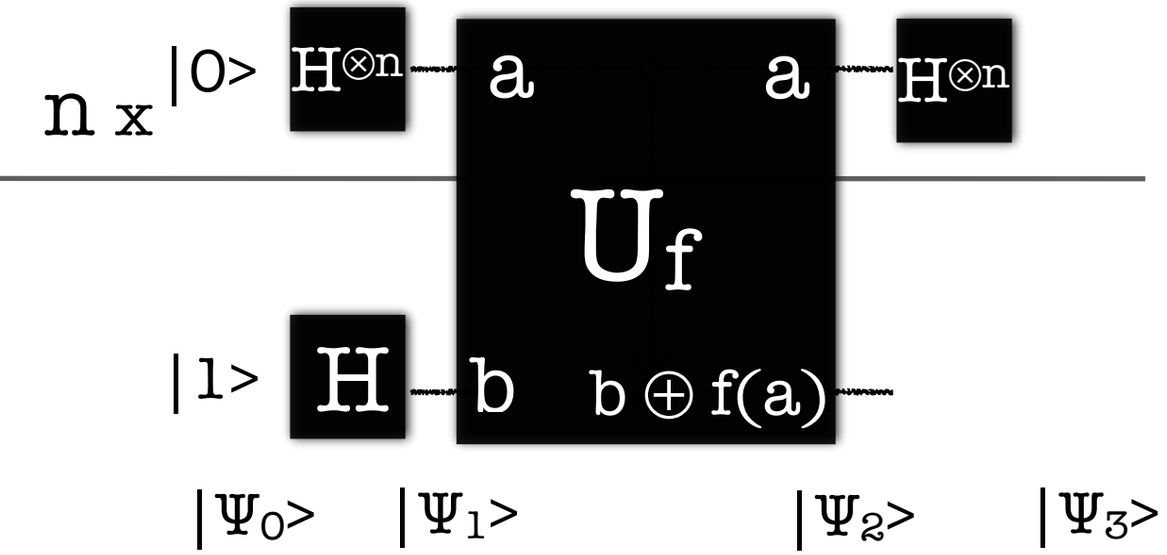


Alice applies H transform on the query register

$$|\Psi_3\rangle = \sum_z \sum_x \frac{(-1)^{x \bullet z + f(x)} |z\rangle}{2^n} \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right]$$

Alice observes the query register …

Deutsch-Jozsa



Alice applies H transform on the query register

$$|\Psi_3\rangle = \sum_z \sum_x \frac{(-1)^{x \bullet z + f(x)} |z>}{2^n} [\frac{|0> - |1>}{\sqrt{2}}]$$

Alice observes the query register ...

The probability amplitude of state $|0>^{\otimes n}$ $[\sum_x \frac{(-1)^{f(x)}}{2^n}]^2$

f(x) constant: prob.(00...0...0) = 1

f(x) balanced: prob.(00...0...0) = 0

$|z> = |0\,0 \ldots 0>$

$z_1 z_2 z_3 \ldots z_n$

Probability of observing $\underbrace{|000\ldots0\rangle}_{n} > 2$

$$[\sum_x \frac{(-1)^{f(x)}}{2^n}]^2 =$$

f(x) const:

$\star \rightarrow 1$: $\left(\frac{2^n}{2^n}\right)^2$

0: $\left(\frac{2^n}{2^n}\right)^2$

$\star\star$

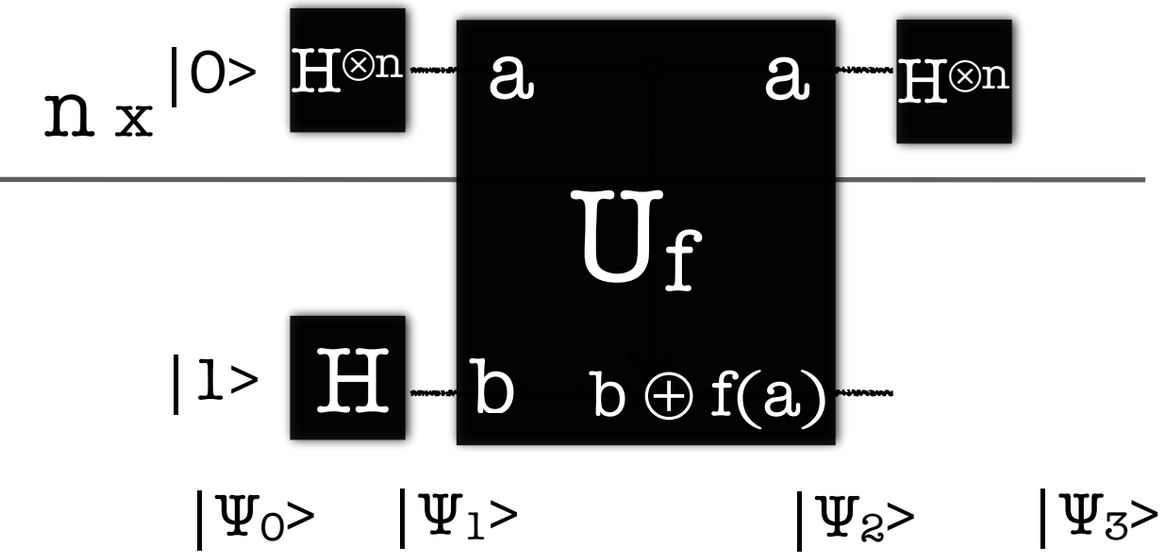
$= 1$

f(x) balanced:

$\left(\frac{2^n/2 - 2^n/2}{2^n}\right)^2$

$= 0$

Deutsch-Jozsa



Alice applies H transform on the query register

$$|\Psi_3\rangle = \sum_z \sum_x \frac{(-1)^{x \bullet z + f(x)} |z\rangle}{2^n} \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]$$

Alice observes the query register ...

The probability amplitude of state $|0\rangle^{\otimes n}$ $\left[\sum_x \frac{(-1)^{f(x)}}{2^n}\right]^2$

$|2\rangle = |00\ldots0\rangle$

f(x) constant: prob.(00...0...0) = 1

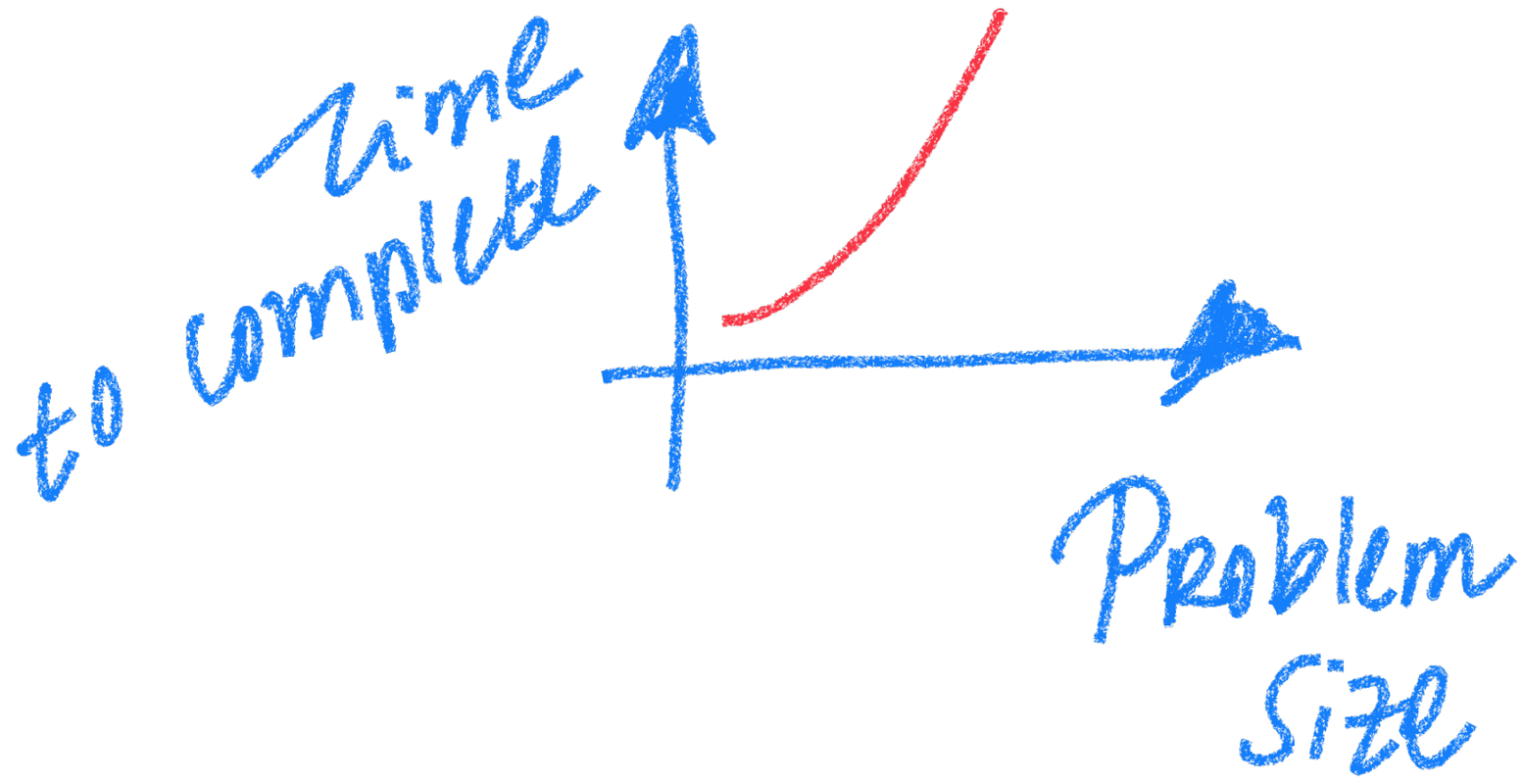
f(x) balanced: prob.(00...0...0) = 0

$f(x)$ const : $P(\text{observing all 0s}) = 1$

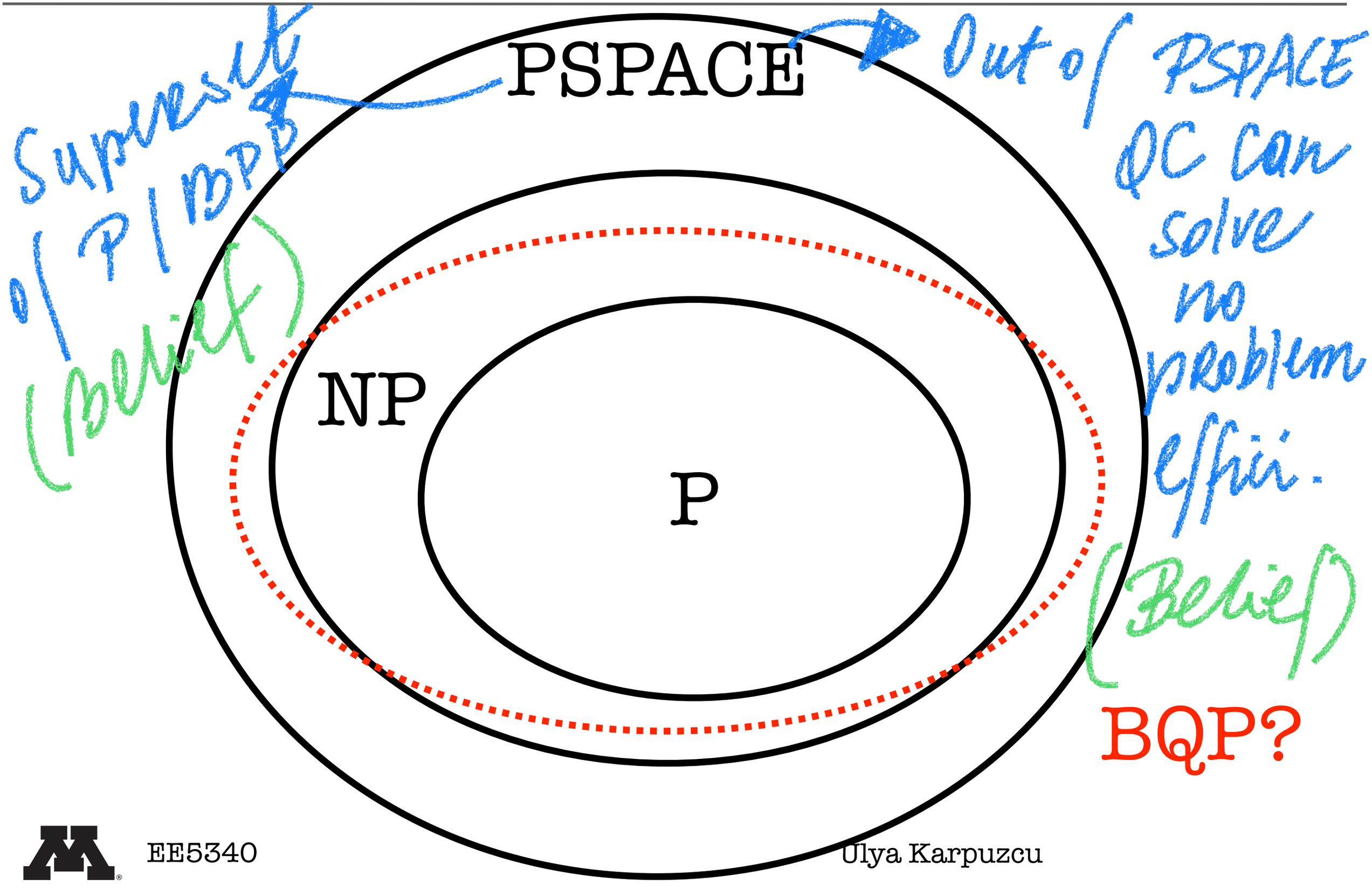
$f(x)$ balanced : $\quad -\,''\,- \quad = 0$

No ambiguity!

Functionality vs. Efficiency



How powerful are quantum computers?



Proxy for complexity: time or space

storage

grows as a function of (#bits or #qbits)

problem size

input dataset size

Polynomial or slower ⇒ "efficient"

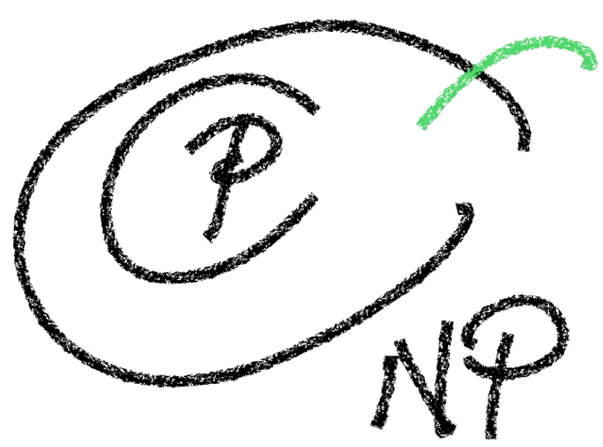
[Classes of problems]

P vs. NP

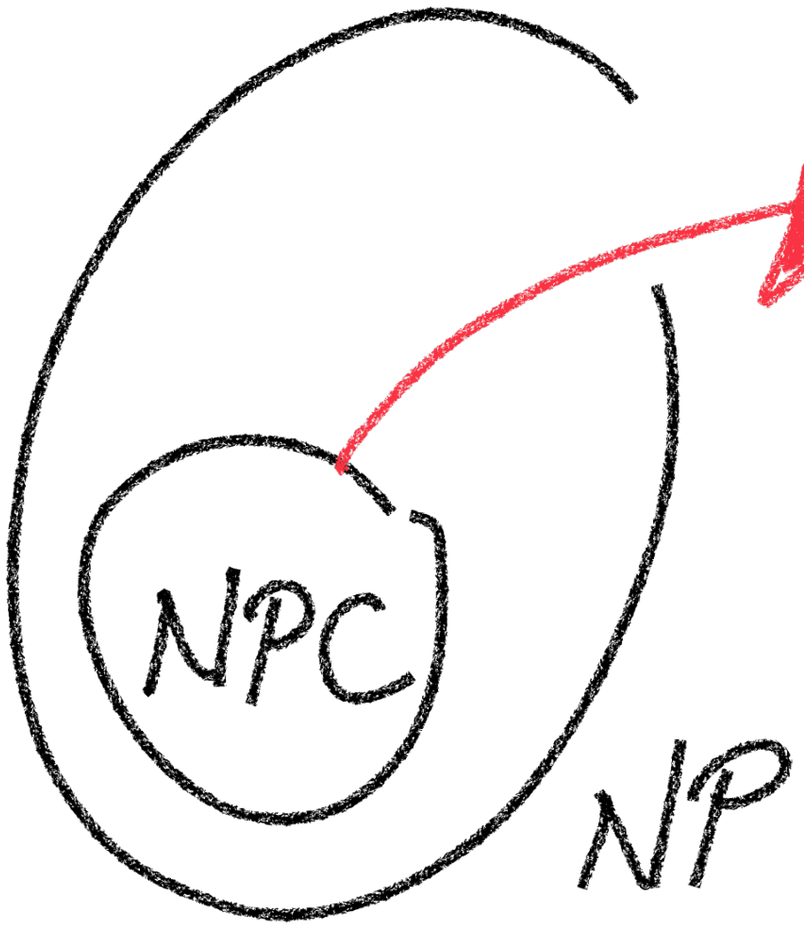
can't solve efficiently but chk solutions efficiently

P = NP ? not known

known:



how big is the diff.?



can reduce to each other in (at most) polynomial time

$P \neq NP$ if proven: no NP can be solved efficiently on a classical machine...

We don't know whether QC can solve all NP efficiently

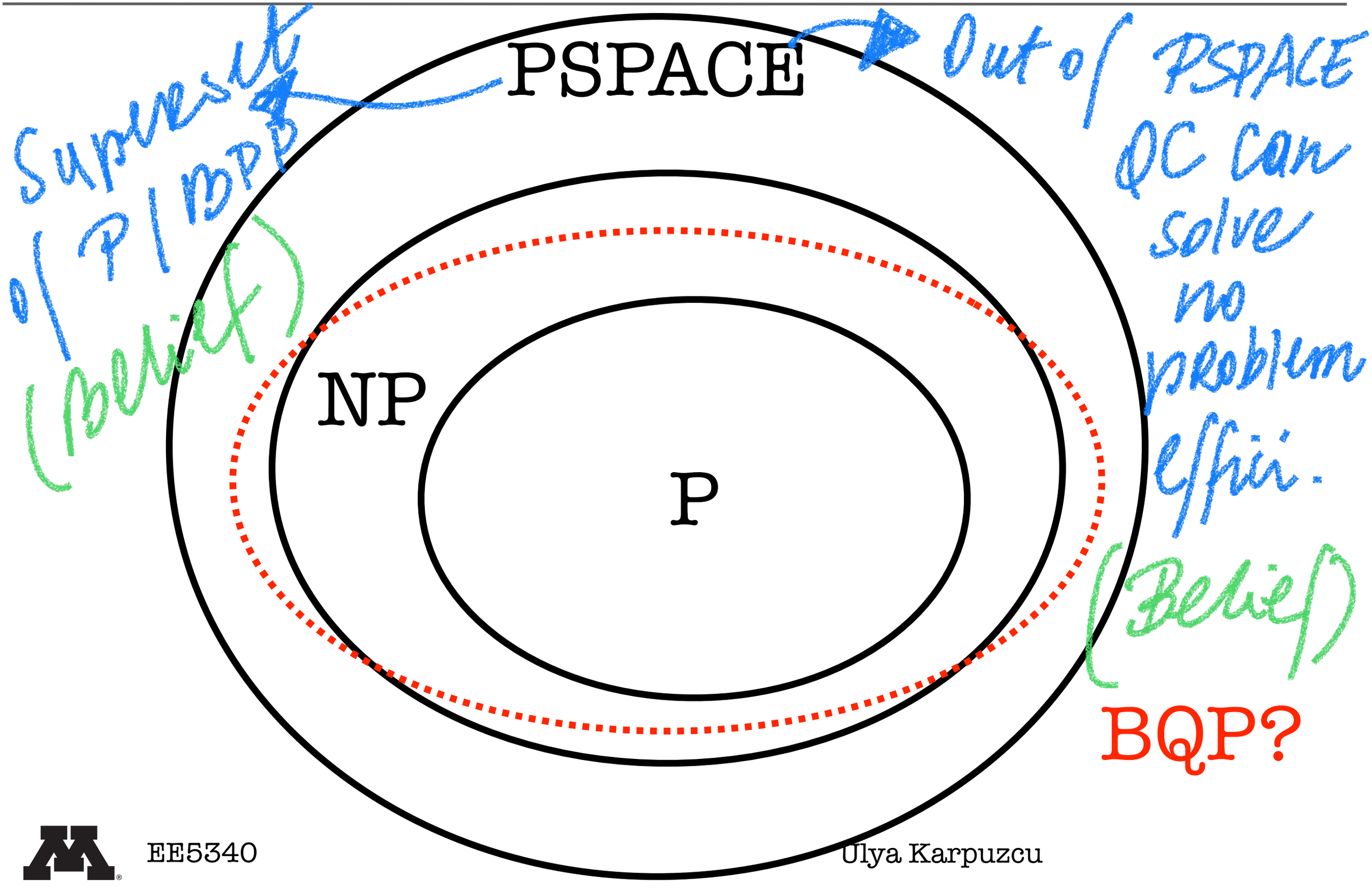
We know that QC can solve some NP efficiently

BPP: Generalization of P for probabilistic algorithms

↓

BQP: Quantum Equivalent (contains P, BPP)

How powerful are quantum computers?



Bibliography

- Feynman Lectures on Computation, Chapter V
- Metodi et al., Quantum Computing for Computer Architects
- Nielsen and Chuang, Quantum Computation and Quantum Information

EE5340

INTRODUCTION TO QUANTUM COMPUTING AND PHYSICAL BASICS OF COMPUTING

Quantum Algorithms

Ulya Karpuzcu