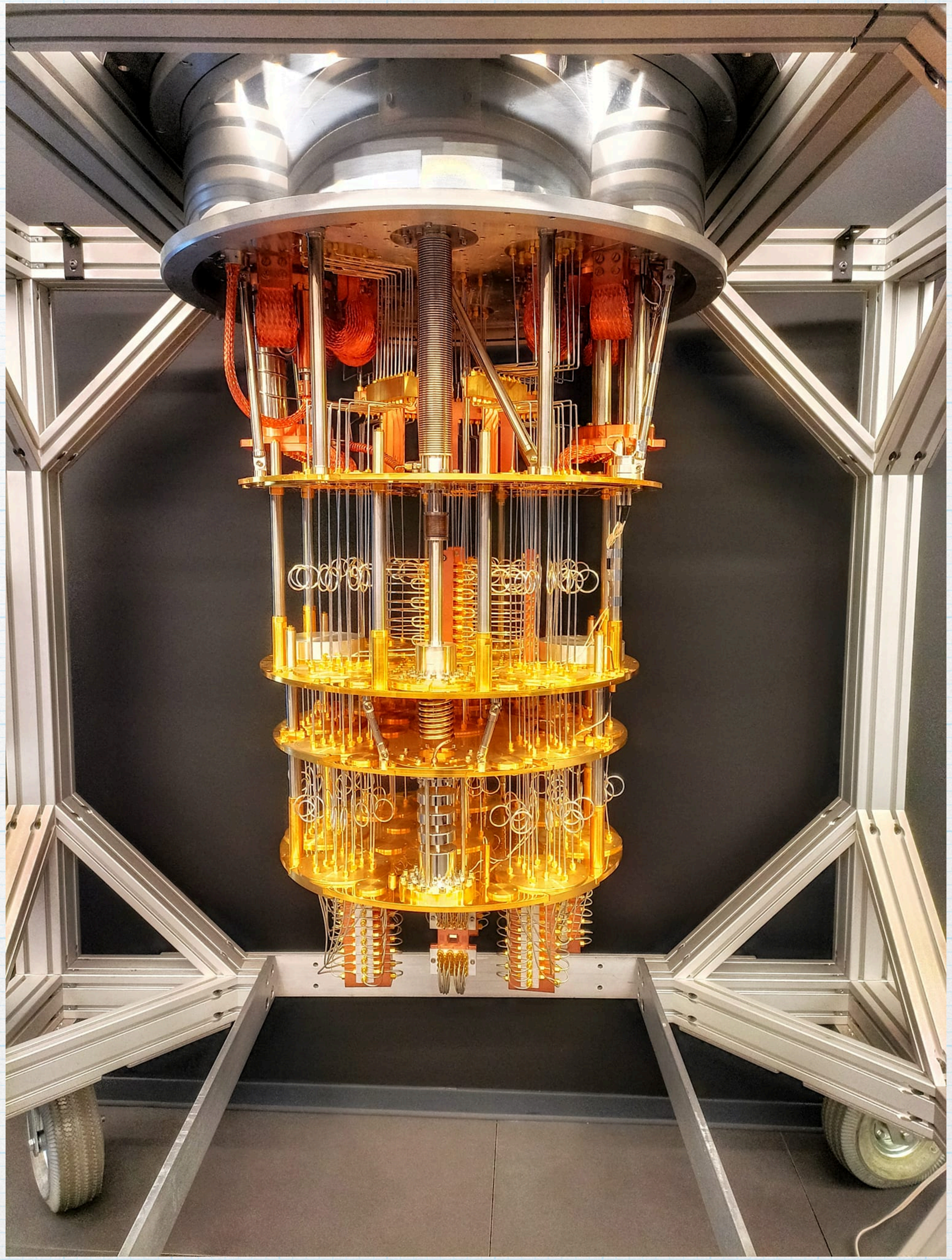
Find the secret dot product string

(Bernstein Vazirani Algorithm)

Ritajit Majumdar

Senior Research Fellow,
Indian Statistical Institute, Kolkata
Qiskit Advocate
Former Fulbright-Nehru Fellow at IBM Quantum



Winter School on Quantum Computing, 2022
IISER Kolkata

Secret dot product string

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

Promise: $f(x) = s \,.\, x$

Task: Find s

$$s\,.\,x = s_{n-1}\,.\,x_{n-1} + s_{n-2}\,.\,x_{n-2} + \ldots + s_0\,.\,x_0$$

$$x \;—\; \boxed{f} \;—\; f(x)$$

Example: Take n = 3 and s = 011

$$x = 000 \Rightarrow f(x) = 0.0 + 0.1 + 0.1 = 0$$

$$x = 001 \Rightarrow f(x) = 0.0 + 0.1 + 1.1 = 1$$

$$x = 010 \Rightarrow f(x) = 0.0 + 1.1 + 0.1 = 1$$

$$x = 011 \Rightarrow f(x) = 0.0 + 1.1 + 1.1 = 0$$

$$x = 100 \Rightarrow f(x) = 1.0 + 0.1 + 0.1 = 0$$

$$x = 101 \Rightarrow f(x) = 1.0 + 0.1 + 1.1 = 1$$

$$x = 110 \Rightarrow f(x) = 1.0 + 1.1 + 0.1 = 1$$

$$x = 111 \Rightarrow f(x) = 1.0 + 1.1 + 1.1 = 0$$

Classical approach

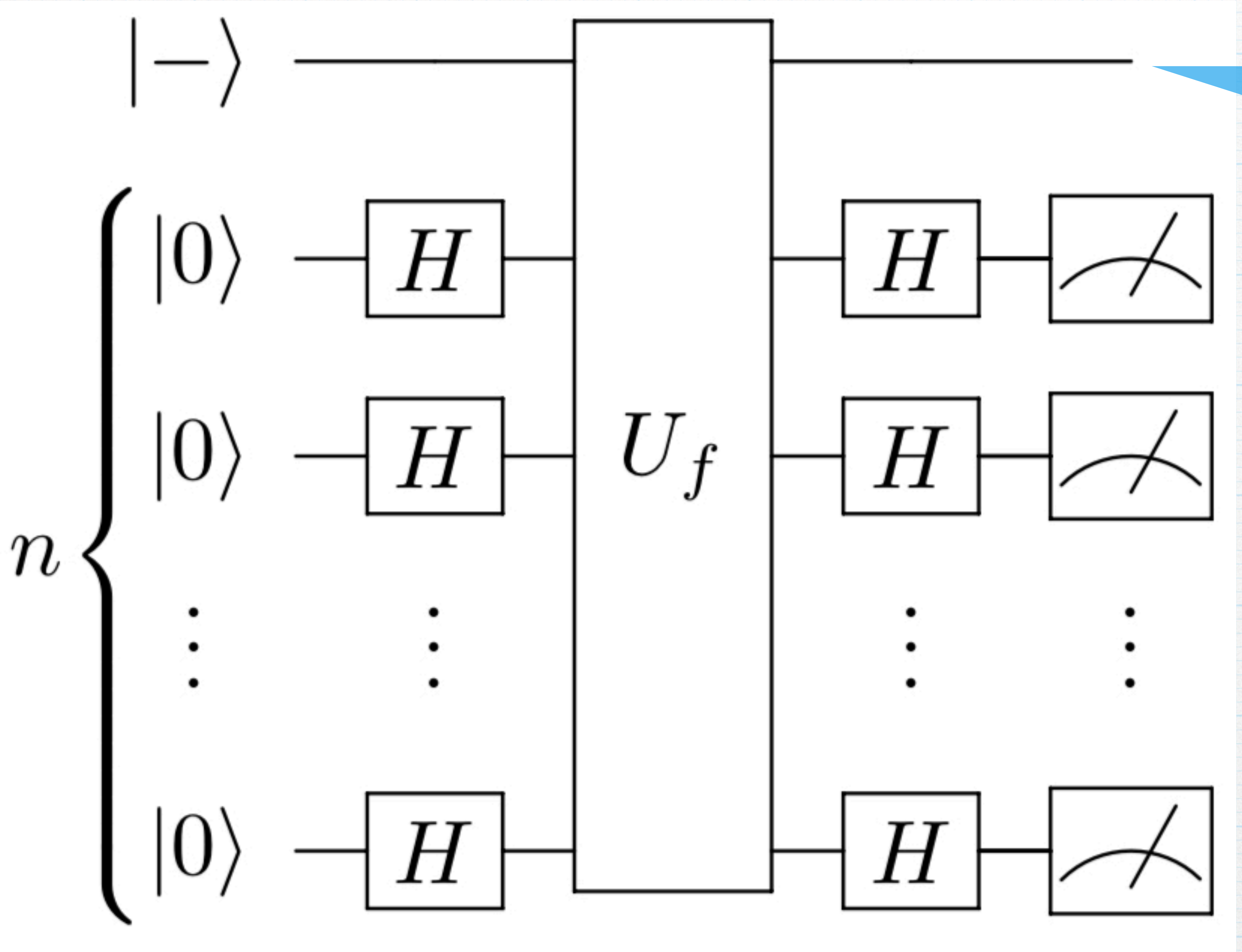
$s = 011 \quad s[0] = 0,\ s[1] = 1,\ s[2] = 1$

$$\left.\begin{array}{l} I_1 : 001 \Rightarrow f(I_1) = 1 \Rightarrow s[2] = 1 \\[1em] I_2 : 010 \Rightarrow f(I_2) = 1 \Rightarrow s[1] = 1 \\[1em] I_3 : 100 \Rightarrow f(I_3) = 0 \Rightarrow s[0] = 0 \end{array}\right\} \text{3 queries are necessary}$$

For an n bit string s, n queries are necessary

Complexity: # queries, i.e. $\mathcal{O}(n)$

Quantum approach



Ancilla qubit

Q: How to create U_f from f?

Phase kickback

General notion: Bit and phase values are independent of each other

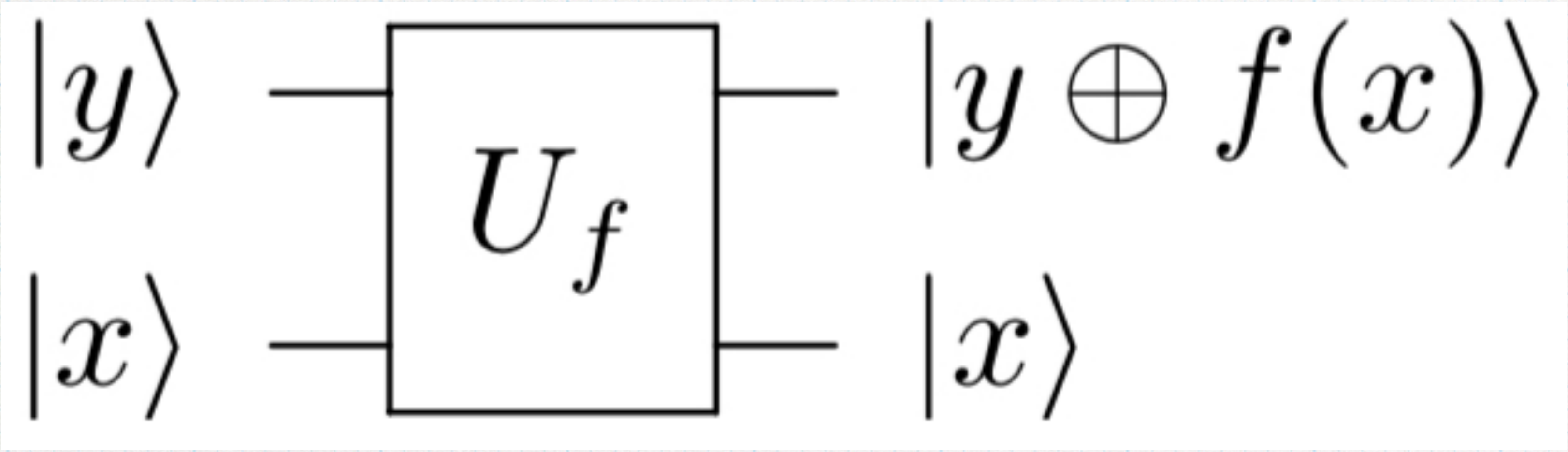
$$|q>: a|0> + b|1>$$

$$X|q> = a|1> + b|0>$$

$$Z|q> = a|0> - b|1>$$

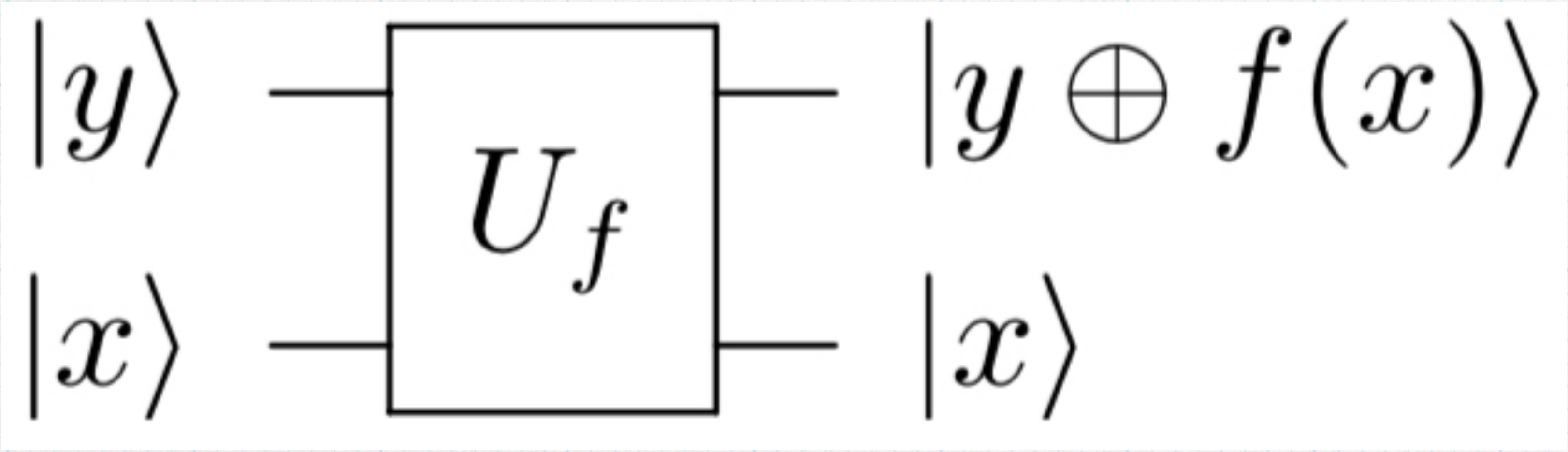
$$Y = iZX$$

Phase kickback (contd.)



Take $|y> = |->$

Phase kickback (contd.)



Take $|y> = |->$

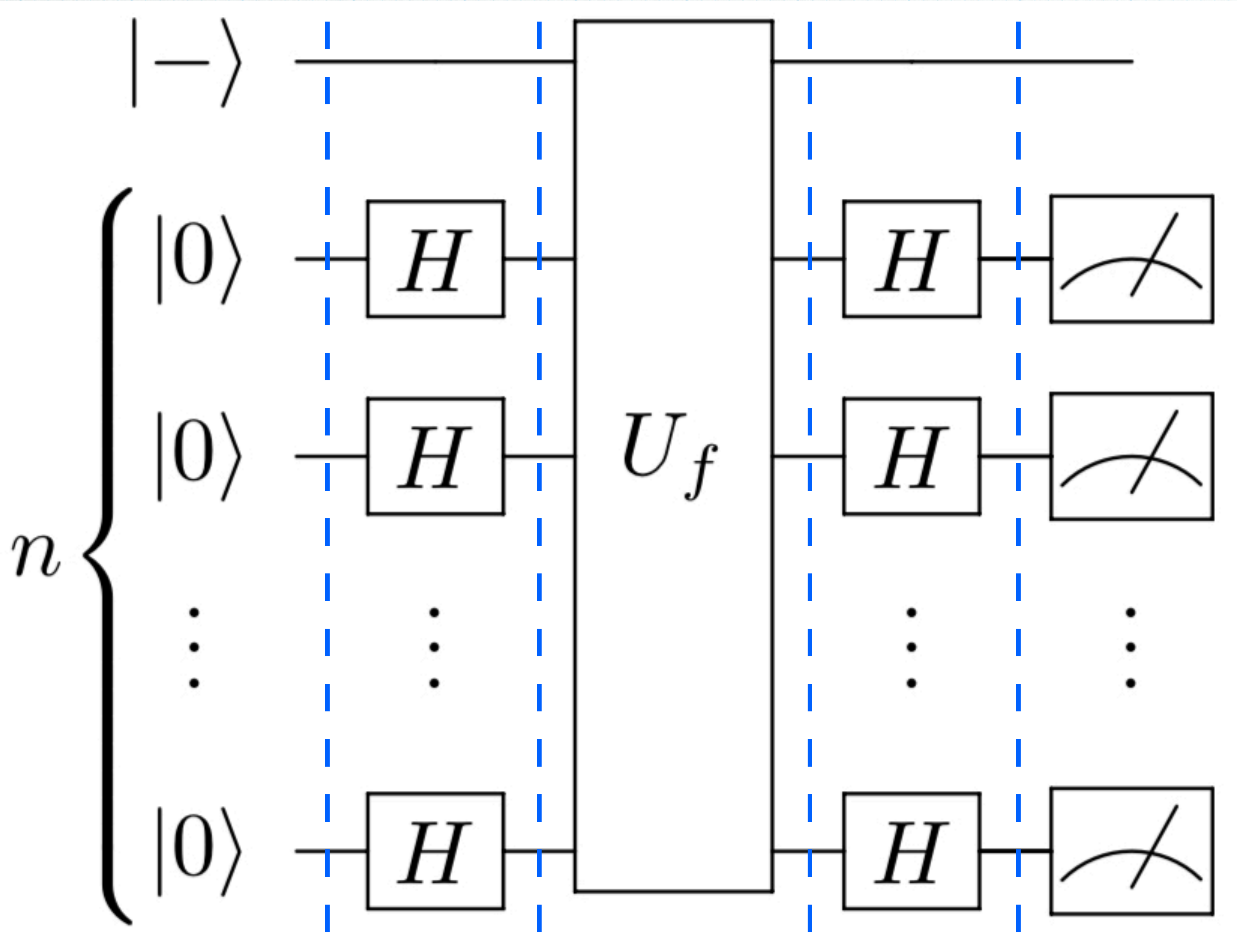
$$|x> \overset{U_f}{\rightarrow} (-1)^{f(x)}|x>$$

$$\begin{aligned}
|x\rangle|-\rangle &= |x\rangle\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \\
&= \frac{1}{\sqrt{2}}(|x\rangle|0\rangle - |x\rangle|1\rangle) \\
&\xrightarrow{U_f} \frac{1}{\sqrt{2}}(|x\rangle|0 \oplus f(x)\rangle - |x\rangle|1 \oplus f(x)\rangle) \\
&= \begin{cases} \frac{1}{\sqrt{2}}(|x\rangle|0\rangle - |x\rangle|1\rangle), & f(x) = 0 \\ \frac{1}{\sqrt{2}}(|x\rangle|1\rangle - |x\rangle|0\rangle), & f(x) = 1 \end{cases} \\
&= \begin{cases} |x\rangle|-\rangle, & f(x) = 0 \\ -|x\rangle|-\rangle, & f(x) = 1 \end{cases} \\
&= (-1)^{f(x)}|x\rangle|-\rangle.
\end{aligned}$$

Action of Hadamard gate

$$H^{\otimes n} | a > = \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} (-1)^{a.x} | x >$$

Bernstein Vazirani algorithm

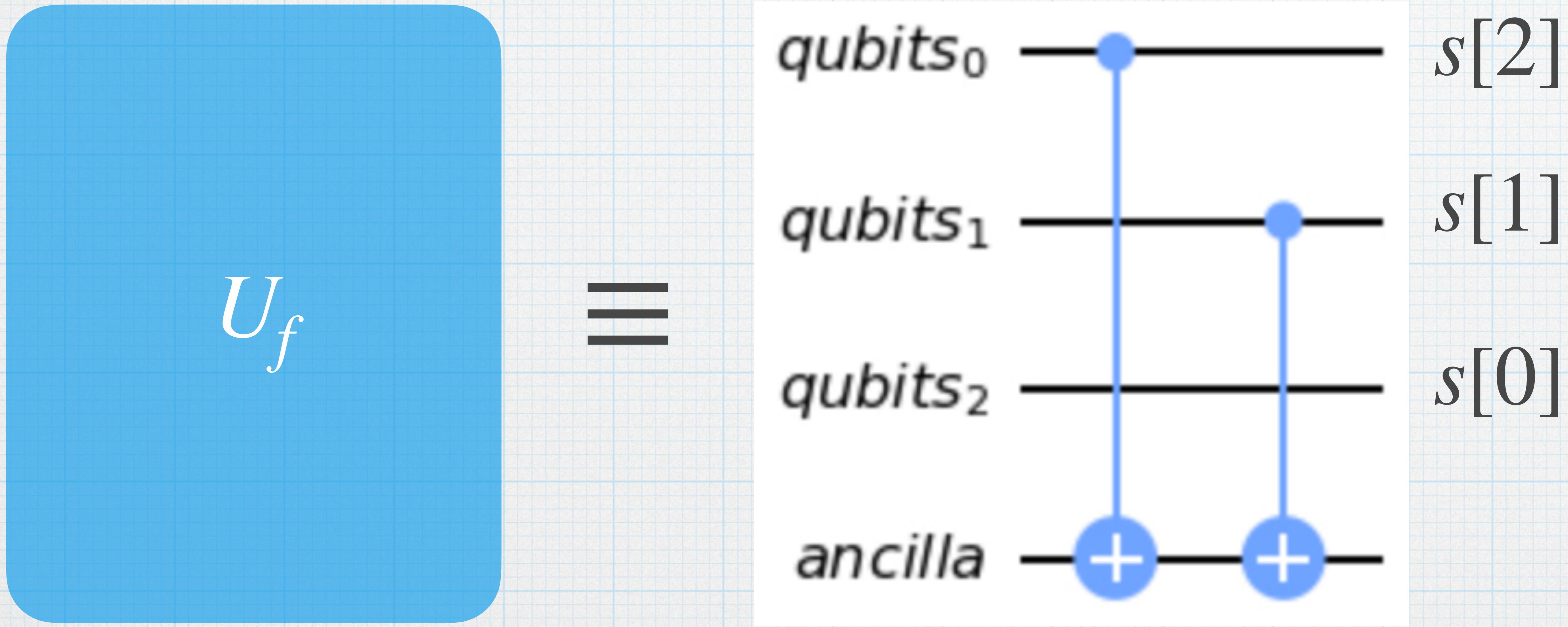


Example

$n = 2 \quad s = 11$

How to prepare U_f

$s = 011 \quad s[0] = 0,\ s[1] = 1,\ s[2] = 1$



Hands on problems

Q1. Write a qiskit code for the Bernstein Vazirani problem where the secret string s = 11101101. Run it on a simulator and show the outcome.

Q2. Write a generalized function for Bernstein Vazirani problem that takes as input the number of qubits n and the secret string s, generates the appropriate quantum circuit, run it on a simulator and show the outcome.

Bonus problem

Run your code in a real quantum device

```
# Load our saved IBMQ accounts and get the least busy backend device with less than or equal to 5 qubits
IBMQ.load_account()
provider = IBMQ.get_provider(hub='ibm-q')
provider.backends()
backend = least_busy(provider.backends(filters=lambda x: x.configuration().n_qubits <= 5 and
                                   x.configuration().n_qubits >= 2 and
                                   not x.configuration().simulator and x.status().operational==True))
print("least busy backend: ", backend)
```

```
from qiskit import IBMQ
IBMQ.save_account(TOKEN)
```