# Today's outline - February 15, 2022

# Today's outline - February 15, 2022

- Deutch's problem

# Today's outline - February 15, 2022

- Deutch's problem
- Quantum subroutines

# Today's outline - February 15, 2022

- Deutch's problem
- Quantum subroutines
- State-dependent phase shift

# Today's outline - February 15, 2022

- Deutch's problem
- Quantum subroutines
- State-dependent phase shift
- State-dependent amplitude shift

# Today's outline - February 15, 2022

- Deutch's problem
- Quantum subroutines
- State-dependent phase shift
- State-dependent amplitude shift

Reading Assignment: Chapter 7.5-7.7

# Today's outline - February 15, 2022

- Deutch's problem
- Quantum subroutines
- State-dependent phase shift
- State-dependent amplitude shift

Reading Assignment: Chapter 7.5-7.7

Homework Assignment #05:
Chapter 7:1,3,4
due Thursday, February 24, 2022

# Deutch's algorithm

The first truly quantum algorithm was described by Deutch in 1985 and demonstrated that quantum computation could outperform classical computation

# Deutch's algorithm

The first truly quantum algorithm was described by Deutch in 1985 and demonstrated that quantum computation could outperform classical computation

This black box problem determines whether a function, $f$ is constant or balanced

# Deutch's algorithm

The first truly quantum algorithm was described by Deutch in 1985 and demonstrated that quantum computation could outperform classical computation

This black box problem determines whether a function, $f$ is constant or balanced

$$f(0) \longrightarrow 0; \quad f(1) \longrightarrow 0$$

# Deutch's algorithm

The first truly quantum algorithm was described by Deutch in 1985 and demonstrated that quantum computation could outperform classical computation

This black box problem determines whether a function, $f$ is constant or balanced

$$f(0) \longrightarrow 0; \quad f(1) \longrightarrow 0$$
$$f(0) \longrightarrow 1; \quad f(1) \longrightarrow 1$$

# Deutch's algorithm

The first truly quantum algorithm was described by Deutch in 1985 and demonstrated that quantum computation could outperform classical computation

This black box problem determines whether a function, $f$ is constant or balanced

$$f(0) \longrightarrow 0; \quad f(1) \longrightarrow 0$$

$$f(0) \longrightarrow 1; \quad f(1) \longrightarrow 1$$

$$f(0) \longrightarrow 0; \quad f(1) \longrightarrow 1$$

# Deutch's algorithm

The first truly quantum algorithm was described by Deutch in 1985 and demonstrated that quantum computation could outperform classical computation

This black box problem determines whether a function, $f$ is constant or balanced

$$f(0) \longrightarrow 0; \quad f(1) \longrightarrow 0$$
$$f(0) \longrightarrow 1; \quad f(1) \longrightarrow 1$$
$$f(0) \longrightarrow 0; \quad f(1) \longrightarrow 1$$
$$f(0) \longrightarrow 1; \quad f(1) \longrightarrow 0$$

# Deutch's algorithm

The first truly quantum algorithm was described by Deutch in 1985 and demonstrated that quantum computation could outperform classical computation

This black box problem determines whether a function, $f$ is constant or balanced

$$f(0) \longrightarrow 0; \quad f(1) \longrightarrow 0$$
$$f(0) \longrightarrow 1; \quad f(1) \longrightarrow 1$$
$$f(0) \longrightarrow 0; \quad f(1) \longrightarrow 1$$
$$f(0) \longrightarrow 1; \quad f(1) \longrightarrow 0$$

Classically, this would require two calls to the black box, one for each value of the input bit, but with a quantum algorithm, only one call is necessary

# Deutch's algorithm

The first truly quantum algorithm was described by Deutch in 1985 and demonstrated that quantum computation could outperform classical computation

$$f(0) \longrightarrow 0; \quad f(1) \longrightarrow 0$$
$$f(0) \longrightarrow 1; \quad f(1) \longrightarrow 1$$
$$f(0) \longrightarrow 0; \quad f(1) \longrightarrow 1$$
$$f(0) \longrightarrow 1; \quad f(1) \longrightarrow 0$$

This black box problem determines whether a function, $f$ is constant or balanced

Classically, this would require two calls to the black box, one for each value of the input bit, but with a quantum algorithm, only one call is necessary

Implementation requires a two-qubit unitary transformation $U_f|x\rangle|y\rangle \rightarrow |x\rangle|y \oplus f(x)\rangle$

# Deutch's algorithm

The first truly quantum algorithm was described by Deutch in 1985 and demonstrated that quantum computation could outperform classical computation

This black box problem determines whether a function, $f$ is constant or balanced

$$f(0) \longrightarrow 0; \quad f(1) \longrightarrow 0$$
$$f(0) \longrightarrow 1; \quad f(1) \longrightarrow 1$$
$$f(0) \longrightarrow 0; \quad f(1) \longrightarrow 1$$
$$f(0) \longrightarrow 1; \quad f(1) \longrightarrow 0$$

Classically, this would require two calls to the black box, one for each value of the input bit, but with a quantum algorithm, only one call is necessary

Implementation requires a two-qubit unitary transformation $U_f|x\rangle|y\rangle \to |x\rangle|y \oplus f(x)\rangle$

$$U_f : |x\rangle|0\rangle \longrightarrow |x\rangle|f(x)\rangle$$

# Deutch's algorithm

The first truly quantum algorithm was described by Deutch in 1985 and demonstrated that quantum computation could outperform classical computation

This black box problem determines whether a function, $f$ is constant or balanced

$$f(0) \longrightarrow 0; \quad f(1) \longrightarrow 0$$
$$f(0) \longrightarrow 1; \quad f(1) \longrightarrow 1$$
$$f(0) \longrightarrow 0; \quad f(1) \longrightarrow 1$$
$$f(0) \longrightarrow 1; \quad f(1) \longrightarrow 0$$

Classically, this would require two calls to the black box, one for each value of the input bit, but with a quantum algorithm, only one call is necessary

Implementation requires a two-qubit unitary transformation $U_f|x\rangle|y\rangle \rightarrow |x\rangle|y \oplus f(x)\rangle$

$$U_f : |x\rangle|0\rangle \longrightarrow |x\rangle|f(x)\rangle$$

Applying $U_f$ to two qubits in the Hadamard basis gives

# Deutch's algorithm

The first truly quantum algorithm was described by Deutch in 1985 and demonstrated that quantum computation could outperform classical computation

$$f(0) \longrightarrow 0; \quad f(1) \longrightarrow 0$$
$$f(0) \longrightarrow 1; \quad f(1) \longrightarrow 1$$
$$f(0) \longrightarrow 0; \quad f(1) \longrightarrow 1$$
$$f(0) \longrightarrow 1; \quad f(1) \longrightarrow 0$$

This black box problem determines whether a function, $f$ is constant or balanced

Classically, this would require two calls to the black box, one for each value of the input bit, but with a quantum algorithm, only one call is necessary

Implementation requires a two-qubit unitary transformation $U_f|x\rangle|y\rangle \rightarrow |x\rangle|y \oplus f(x)\rangle$

$$U_f : |x\rangle|0\rangle \longrightarrow |x\rangle|f(x)\rangle$$

Applying $U_f$ to two qubits in the Hadamard basis gives

$$U_f|+-\rangle = U_f\left[\tfrac{1}{2}\big(|0\rangle + |1\rangle\big)\big(|0\rangle - |1\rangle\big)\right]$$

# Deutch's algorithm

The first truly quantum algorithm was described by Deutch in 1985 and demonstrated that quantum computation could outperform classical computation

This black box problem determines whether a function, $f$ is constant or balanced

$$f(0) \longrightarrow 0; \quad f(1) \longrightarrow 0$$
$$f(0) \longrightarrow 1; \quad f(1) \longrightarrow 1$$
$$f(0) \longrightarrow 0; \quad f(1) \longrightarrow 1$$
$$f(0) \longrightarrow 1; \quad f(1) \longrightarrow 0$$

Classically, this would require two calls to the black box, one for each value of the input bit, but with a quantum algorithm, only one call is necessary

Implementation requires a two-qubit unitary transformation $U_f|x\rangle|y\rangle \rightarrow |x\rangle|y \oplus f(x)\rangle$

$$U_f : |x\rangle|0\rangle \longrightarrow |x\rangle|f(x)\rangle$$

Applying $U_f$ to two qubits in the Hadamard basis gives

$$U_f|+-\rangle = U_f\left[\tfrac{1}{2}\big(|0\rangle + |1\rangle\big)\big(|0\rangle - |1\rangle\big)\right]$$
$$= \tfrac{1}{2}\Big[|0\rangle\big(|0 \oplus f(0)\rangle - |1 \oplus f(0)\rangle\big) + |1\rangle\big(|0 \oplus f(1)\rangle - |1 \oplus f(1)\rangle\big)\Big]$$

# Deutch's algorithm

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

# Deutch's algorithm

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

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

# Deutch's algorithm

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

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

when $f(x) = 0$ then

# Deutch's algorithm

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

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

when $f(x) = 0$ then $\frac{1}{\sqrt{2}}\big(|0\oplus f(x)\rangle - |1\oplus f(x)\rangle\big) \longrightarrow \frac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big) = +|-\rangle$

# Deutch's algorithm

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

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

when $f(x) = 0$ then $\frac{1}{\sqrt{2}}\big(|0 \oplus f(x)\rangle - |1 \oplus f(x)\rangle\big) \longrightarrow \frac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big) = +|-\rangle$

but if $f(x) = 1$ then

# Deutch's algorithm

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

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

when $f(x) = 0$ then $\frac{1}{\sqrt{2}}\big(|0\oplus f(x)\rangle - |1\oplus f(x)\rangle\big) \longrightarrow \frac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big) = +|-\rangle$

but if $f(x) = 1$ then $\frac{1}{\sqrt{2}}\big(|0\oplus f(x)\rangle - |1\oplus f(x)\rangle\big) \longrightarrow \frac{1}{\sqrt{2}}\big(|1\rangle - |0\rangle\big) = -|-\rangle$

# Deutch's algorithm

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

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

when $f(x)=0$ then $\quad \frac{1}{\sqrt{2}}\big(|0\oplus f(x)\rangle - |1\oplus f(x)\rangle\big) \longrightarrow \frac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big) = +|-\rangle$

but if $f(x)=1$ then $\quad \frac{1}{\sqrt{2}}\big(|0\oplus f(x)\rangle - |1\oplus f(x)\rangle\big) \longrightarrow \frac{1}{\sqrt{2}}\big(|1\rangle - |0\rangle\big) = -|-\rangle$

Thus, in a more compact notation, we write

# Deutch's algorithm

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

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

when $f(x)=0$ then $\qquad \frac{1}{\sqrt{2}}\big(|0\oplus f(x)\rangle - |1\oplus f(x)\rangle\big) \longrightarrow \frac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big) = +|-\rangle$

but if $f(x)=1$ then $\qquad \frac{1}{\sqrt{2}}\big(|0\oplus f(x)\rangle - |1\oplus f(x)\rangle\big) \longrightarrow \frac{1}{\sqrt{2}}\big(|1\rangle - |0\rangle\big) = -|-\rangle$

Thus, in a more compact notation, we write

$$U_f|+\rangle|-\rangle = \frac{1}{\sqrt{2}}\sum_{x=0}^{1}(-1)^{f(x)}|x\rangle|-\rangle$$

# Deutch's algorithm

$$U_f|+\rangle|-\rangle = \frac{1}{2}\Big[|0\rangle\big(|0\oplus f(0)\rangle - |1\oplus f(0)\rangle\big) + |1\rangle\big(|0\oplus f(1)\rangle - |1\oplus f(1)\rangle\big)\Big]$$
$$= \frac{1}{2}\sum_{x=0}^{1}|x\rangle\big(|0\oplus f(x)\rangle - |1\oplus f(x)\rangle\big)$$

when $f(x) = 0$ then $\frac{1}{\sqrt{2}}\big(|0\oplus f(x)\rangle - |1\oplus f(x)\rangle\big) \longrightarrow \frac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big) = +|-\rangle$

but if $f(x) = 1$ then $\frac{1}{\sqrt{2}}\big(|0\oplus f(x)\rangle - |1\oplus f(x)\rangle\big) \longrightarrow \frac{1}{\sqrt{2}}\big(|1\rangle - |0\rangle\big) = -|-\rangle$

Thus, in a more compact notation, we write

$$U_f|+\rangle|-\rangle = \frac{1}{\sqrt{2}}\sum_{x=0}^{1}(-1)^{f(x)}|x\rangle|-\rangle$$

For $f(x)$ constant, both terms pick up the same phase shift and the state is $|+\rangle|-\rangle$

# Deutch's algorithm

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

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

when $f(x) = 0$ then $\frac{1}{\sqrt{2}}\big(|0\oplus f(x)\rangle - |1\oplus f(x)\rangle\big) \longrightarrow \frac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big) = +|-\rangle$

but if $f(x) = 1$ then $\frac{1}{\sqrt{2}}\big(|0\oplus f(x)\rangle - |1\oplus f(x)\rangle\big) \longrightarrow \frac{1}{\sqrt{2}}\big(|1\rangle - |0\rangle\big) = -|-\rangle$

Thus, in a more compact notation, we write

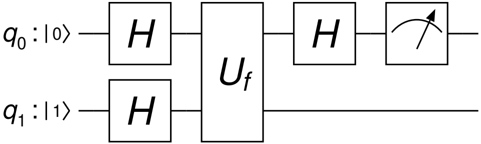
$$U_f|+\rangle|-\rangle = \frac{1}{\sqrt{2}}\sum_{x=0}^{1}(-1)^{f(x)}|x\rangle|-\rangle$$

For $f(x)$ constant, both terms pick up the same phase shift and the state is $|+\rangle|-\rangle$

For $f(x)$ balanced, only one term picks up a phase shift, giving a result of $|-\rangle|-\rangle$

# Deutch's algorithm

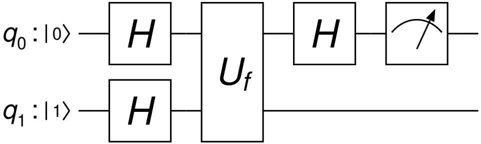
As a quantum circuit, Deutch's algorithm is implemented by



# Deutch's algorithm

As a quantum circuit, Deutch's algorithm is implemented by

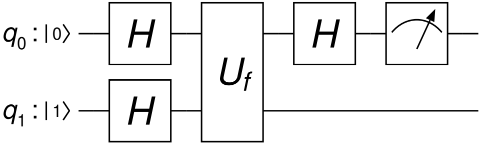
1. prepare two qubits: $q_0 = |0\rangle$ and $q_1 = |1\rangle$



# Deutch's algorithm

As a quantum circuit, Deutch's algorithm is implemented by

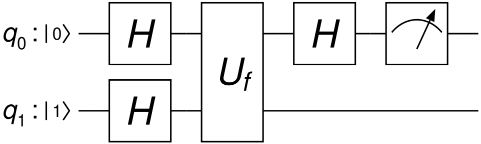
1. prepare two qubits: $q_0 = |0\rangle$ and $q_1 = |1\rangle$
2. apply the Hadamard transform to each qubit



# Deutch's algorithm

As a quantum circuit, Deutch's algorithm is implemented by

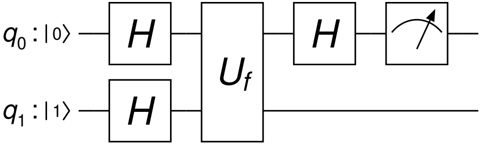
1. prepare two qubits: $q_0 = |0\rangle$ and $q_1 = |1\rangle$
2. apply the Hadamard transform to each qubit
3. apply the black box algorithm



# Deutch's algorithm

As a quantum circuit, Deutch's algorithm is implemented by

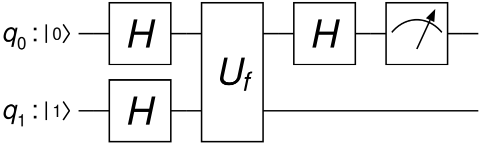
1. prepare two qubits: $q_0 = |0\rangle$ and $q_1 = |1\rangle$
2. apply the Hadamard transform to each qubit
3. apply the black box algorithm
4. apply the Hadamard transform to $q_0$



# Deutch's algorithm

As a quantum circuit, Deutch's algorithm is implemented by

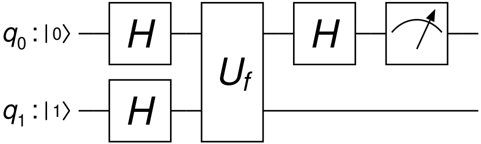
1. prepare two qubits: $q_0 = |0\rangle$ and $q_1 = |1\rangle$
2. apply the Hadamard transform to each qubit
3. apply the black box algorithm
4. apply the Hadamard transform to $q_0$
5. measure $q_0$ and interpret



# Deutch's algorithm

As a quantum circuit, Deutch's algorithm is implemented by

1. prepare two qubits: $q_0 = |0\rangle$ and $q_1 = |1\rangle$
2. apply the Hadamard transform to each qubit
3. apply the black box algorithm
4. apply the Hadamard transform to $q_0$
5. measure $q_0$ and interpret

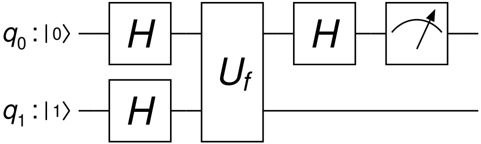



(a) $q_0 = 0 \longrightarrow f(x)$ is constant

# Deutch's algorithm

As a quantum circuit, Deutch's algorithm is implemented by

1. prepare two qubits: $q_0 = |0\rangle$ and $q_1 = |1\rangle$
2. apply the Hadamard transform to each qubit
3. apply the black box algorithm
4. apply the Hadamard transform to $q_0$
5. measure $q_0$ and interpret




(a) $q_0 = 0 \longrightarrow f(x)$ is constant

(b) $q_0 = 1 \longrightarrow f(x)$ is balanced

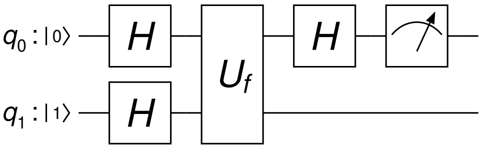
# Deutch's algorithm

As a quantum circuit, Deutch's algorithm is implemented by

1. prepare two qubits: $q_0 = |0\rangle$ and $q_1 = |1\rangle$
2. apply the Hadamard transform to each qubit
3. apply the black box algorithm
4. apply the Hadamard transform to $q_0$
5. measure $q_0$ and interpret

$$q_0 : |0\rangle \rightarrow |+\rangle = \frac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big)$$

$$q_1 : |1\rangle \rightarrow |-\rangle = \frac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big)$$




(a) $q_0 = 0 \longrightarrow f(x)$ is constant

(b) $q_0 = 1 \longrightarrow f(x)$ is balanced

# Deutch's algorithm

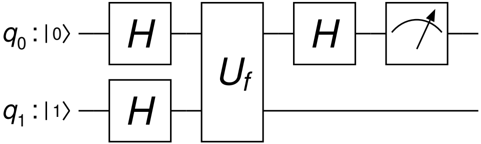
As a quantum circuit, Deutch's algorithm is implemented by

1. prepare two qubits: $q_0 = |0\rangle$ and $q_1 = |1\rangle$
2. apply the Hadamard transform to each qubit
3. apply the black box algorithm
4. apply the Hadamard transform to $q_0$
5. measure $q_0$ and interpret

$$q_0 : |0\rangle \rightarrow |+\rangle = \tfrac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big)$$

$$q_1 : |1\rangle \rightarrow |-\rangle = \tfrac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big)$$

$$U_f|+\rangle|-\rangle = \frac{1}{\sqrt{2}}\sum_{x=0}^{1}(-1)^{f(x)}|x\rangle|-\rangle$$




(a) $q_0 = 0 \longrightarrow f(x)$ is constant

(b) $q_0 = 1 \longrightarrow f(x)$ is balanced

# Deutch's algorithm

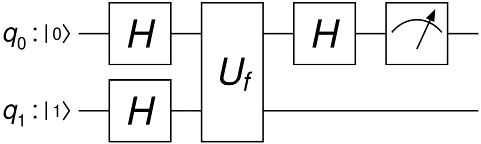
As a quantum circuit, Deutch's algorithm is implemented by

1. prepare two qubits: $q_0 = |0\rangle$ and $q_1 = |1\rangle$
2. apply the Hadamard transform to each qubit
3. apply the black box algorithm
4. apply the Hadamard transform to $q_0$
5. measure $q_0$ and interpret

$$q_0 : |0\rangle \rightarrow |+\rangle = \tfrac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big)$$

$$q_1 : |1\rangle \rightarrow |-\rangle = \tfrac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big)$$

$$U_f|+\rangle|-\rangle = \frac{1}{\sqrt{2}}\sum_{x=0}^{1}(-1)^{f(x)}|x\rangle|-\rangle$$




(a) $q_0 = 0 \longrightarrow f(x)$ is constant

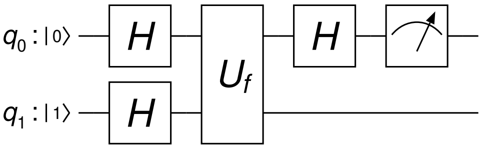
(b) $q_0 = 1 \longrightarrow f(x)$ is balanced

$f(x)$ $U_f|q_0\rangle|q_1\rangle$ $q_0$

# Deutch's algorithm

As a quantum circuit, Deutch's algorithm is implemented by

1. prepare two qubits: $q_0 = |0\rangle$ and $q_1 = |1\rangle$
2. apply the Hadamard transform to each qubit
3. apply the black box algorithm
4. apply the Hadamard transform to $q_0$
5. measure $q_0$ and interpret




(a) $q_0 = 0 \longrightarrow f(x)$ is constant

(b) $q_0 = 1 \longrightarrow f(x)$ is balanced

$$q_0 : |0\rangle \rightarrow |+\rangle = \tfrac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big)$$
$$q_1 : |1\rangle \rightarrow |-\rangle = \tfrac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big)$$

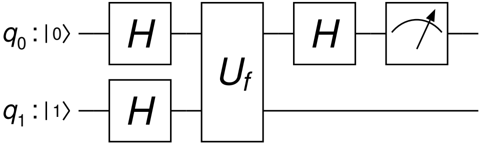
$$U_f|+\rangle|-\rangle = \frac{1}{\sqrt{2}}\sum_{x=0}^{1}(-1)^{f(x)}|x\rangle|-\rangle$$

| $f(x)$ | $U_f\|q_0\rangle\|q_1\rangle$ | $q_0$ |
|---|---|---|
| $f(0) \rightarrow 0;\quad f(1) \rightarrow 0$ | $+\|+\rangle\|-\rangle$ | $\|+\rangle \rightarrow \|0\rangle$ |

# Deutch's algorithm

As a quantum circuit, Deutch's algorithm is implemented by

1. prepare two qubits: $q_0 = |0\rangle$ and $q_1 = |1\rangle$
2. apply the Hadamard transform to each qubit
3. apply the black box algorithm
4. apply the Hadamard transform to $q_0$
5. measure $q_0$ and interpret




(a) $q_0 = 0 \longrightarrow f(x)$ is constant

(b) $q_0 = 1 \longrightarrow f(x)$ is balanced

$$q_0 : |0\rangle \rightarrow |+\rangle = \frac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big)$$

$$q_1 : |1\rangle \rightarrow |-\rangle = \frac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big)$$

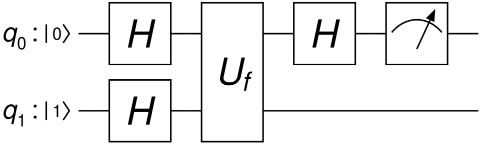
$$U_f|+\rangle|-\rangle = \frac{1}{\sqrt{2}}\sum_{x=0}^{1}(-1)^{f(x)}|x\rangle|-\rangle$$

| $f(x)$ | $U_f\vert q_0\rangle\vert q_1\rangle$ | $q_0$ |
|---|---|---|
| $f(0) \rightarrow 0;\quad f(1) \rightarrow 0$ | $+\vert+\rangle\vert-\rangle$ | $\vert+\rangle \rightarrow \vert 0\rangle$ |
| $f(0) \rightarrow 1;\quad f(1) \rightarrow 1$ | $-\vert+\rangle\vert-\rangle$ | $\vert+\rangle \rightarrow \vert 0\rangle$ |

# Deutch's algorithm

As a quantum circuit, Deutch's algorithm is implemented by

1. prepare two qubits: $q_0 = |0\rangle$ and $q_1 = |1\rangle$
2. apply the Hadamard transform to each qubit
3. apply the black box algorithm
4. apply the Hadamard transform to $q_0$
5. measure $q_0$ and interpret




(a) $q_0 = 0 \longrightarrow f(x)$ is constant

(b) $q_0 = 1 \longrightarrow f(x)$ is balanced

$$q_0 : |0\rangle \rightarrow |+\rangle = \frac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big)$$

$$q_1 : |1\rangle \rightarrow |-\rangle = \frac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big)$$

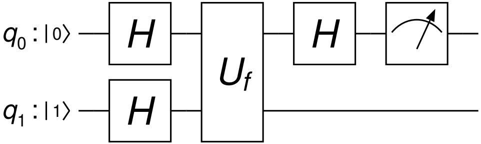
$$U_f|+\rangle|-\rangle = \frac{1}{\sqrt{2}}\sum_{x=0}^{1}(-1)^{f(x)}|x\rangle|-\rangle$$

| $f(x)$ | $U_f\|q_0\rangle\|q_1\rangle$ | $q_0$ |
|---|---|---|
| $f(0) \rightarrow 0;\quad f(1) \rightarrow 0$ | $+\|+\rangle\|-\rangle$ | $\|+\rangle \rightarrow \|0\rangle$ |
| $f(0) \rightarrow 1;\quad f(1) \rightarrow 1$ | $-\|+\rangle\|-\rangle$ | $\|+\rangle \rightarrow \|0\rangle$ |
| $f(0) \rightarrow 0;\quad f(1) \rightarrow 1$ | $+\|-\rangle\|-\rangle$ | $\|-\rangle \rightarrow \|1\rangle$ |

# Deutch's algorithm

As a quantum circuit, Deutch's algorithm is implemented by

1. prepare two qubits: $q_0 = |0\rangle$ and $q_1 = |1\rangle$
2. apply the Hadamard transform to each qubit
3. apply the black box algorithm
4. apply the Hadamard transform to $q_0$
5. measure $q_0$ and interpret




(a) $q_0 = 0 \longrightarrow f(x)$ is constant

(b) $q_0 = 1 \longrightarrow f(x)$ is balanced

$$q_0 : |0\rangle \rightarrow |+\rangle = \frac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big)$$

$$q_1 : |1\rangle \rightarrow |-\rangle = \frac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big)$$

$$U_f|+\rangle|-\rangle = \frac{1}{\sqrt{2}}\sum_{x=0}^{1}(-1)^{f(x)}|x\rangle|-\rangle$$

| $f(x)$ | $U_f\|q_0\rangle\|q_1\rangle$ | $q_0$ |
|---|---|---|
| $f(0) \rightarrow 0;\quad f(1) \rightarrow 0$ | $+\|+\rangle\|-\rangle$ | $\|+\rangle \rightarrow \|0\rangle$ |
| $f(0) \rightarrow 1;\quad f(1) \rightarrow 1$ | $-\|+\rangle\|-\rangle$ | $\|+\rangle \rightarrow \|0\rangle$ |
| $f(0) \rightarrow 0;\quad f(1) \rightarrow 1$ | $+\|-\rangle\|-\rangle$ | $\|-\rangle \rightarrow \|1\rangle$ |
| $f(0) \rightarrow 1;\quad f(1) \rightarrow 0$ | $-\|-\rangle\|-\rangle$ | $\|-\rangle \rightarrow \|1\rangle$ |

# Quantum subroutines

Subroutines are useful in quantum computing as they are for classical computations and often they utilize temporary qubits

# Quantum subroutines

Subroutines are useful in quantum computing as they are for classical computations and often they utilize temporary qubits

These temporary qubits must be uncomputed as leaving them in the system could easily lead to entanglement which would destroy the computation

# Quantum subroutines

Subroutines are useful in quantum computing as they are for classical computations and often they utilize temporary qubits

These temporary qubits must be uncomputed as leaving them in the system could easily lead to entanglement which would destroy the computation

A subroutine that computes $\sum_i \alpha_i |x_i\rangle$ must not compute $\sum_i \alpha_i |x_i\rangle |y_i\rangle$ and simply throw away the qubits that store $|y_i\rangle$ unless it is certain that there is no entanglement with $|x_i\rangle$

# Quantum subroutines

Subroutines are useful in quantum computing as they are for classical computations and often they utilize temporary qubits

These temporary qubits must be uncomputed as leaving them in the system could easily lead to entanglement which would destroy the computation

A subroutine that computes $\sum_i \alpha_i |x_i\rangle$ must not compute $\sum_i \alpha_i |x_i\rangle |y_i\rangle$ and simply throw away the qubits that store $|y_i\rangle$ unless it is certain that there is no entanglement with $|x_i\rangle$

There is no entanglement when we can write

# Quantum subroutines

Subroutines are useful in quantum computing as they are for classical computations and often they utilize temporary qubits

These temporary qubits must be uncomputed as leaving them in the system could easily lead to entanglement which would destroy the computation

A subroutine that computes $\sum_i \alpha_i|x_i\rangle$ must not compute $\sum_i \alpha_i|x_i\rangle|y_i\rangle$ and simply throw away the qubits that store $|y_i\rangle$ unless it is certain that there is no entanglement with $|x_i\rangle$

There is no entanglement when we can write

$$\sum_i \alpha_i|x_i\rangle|y_i\rangle = \Big(\sum_i \alpha_i|x_i\rangle\Big) \otimes |y_i\rangle$$

# Quantum subroutines

Subroutines are useful in quantum computing as they are for classical computations and often they utilize temporary qubits

These temporary qubits must be uncomputed as leaving them in the system could easily lead to entanglement which would destroy the computation

A subroutine that computes $\sum_i \alpha_i|x_i\rangle$ must not compute $\sum_i \alpha_i|x_i\rangle|y_i\rangle$ and simply throw away the qubits that store $|y_i\rangle$ unless it is certain that there is no entanglement with $|x_i\rangle$

There is no entanglement when we can write

$$\sum_i \alpha_i|x_i\rangle|y_i\rangle = \left(\sum_i \alpha_i|x_i\rangle\right) \otimes |y_i\rangle$$

this is possible only when $|y_i\rangle \equiv |y_j\rangle$ for all values of $i$ and $j$

# Quantum subroutines

Subroutines are useful in quantum computing as they are for classical computations and often they utilize temporary qubits

These temporary qubits must be uncomputed as leaving them in the system could easily lead to entanglement which would destroy the computation

A subroutine that computes $\sum_i \alpha_i |x_i\rangle$ must not compute $\sum_i \alpha_i |x_i\rangle |y_i\rangle$ and simply throw away the qubits that store $|y_i\rangle$ unless it is certain that there is no entanglement with $|x_i\rangle$

There is no entanglement when we can write

$$\sum_i \alpha_i |x_i\rangle |y_i\rangle = \left( \sum_i \alpha_i |x_i\rangle \right) \otimes |y_i\rangle$$

this is possible only when $|y_i\rangle \equiv |y_j\rangle$ for all values of $i$ and $j$

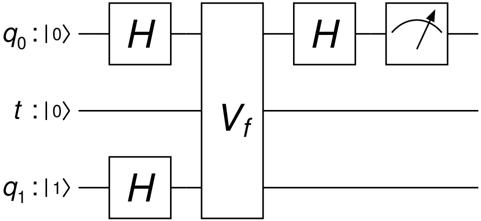
For this reason, it is essential to uncompute $|y_i\rangle$ inside the subroutine before the output qubits are transmitted

# Entanglement problems

Suppose the internal temporary computation, $|t\rangle$ in the Deutch example is made explicit in a transformation $V_f$ such that

# Entanglement problems

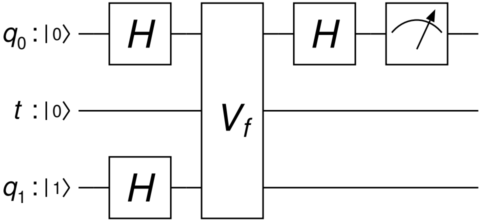
Suppose the internal temporary computation, $|t\rangle$ in the Deutch example is made explicit in a transformation $V_f$ such that



# Entanglement problems

Suppose the internal temporary computation, $|t\rangle$ in the Deutch example is made explicit in a transformation $V_f$ such that

$$V_f : |x, t, y\rangle \rightarrow |x, t \oplus x, y \oplus f(x)\rangle$$

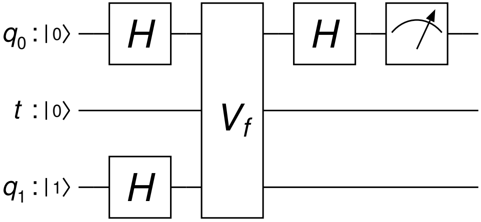


# Entanglement problems

Suppose the internal temporary computation, $|t\rangle$ in the Deutch example is made explicit in a transformation $V_f$ such that

$$V_f : |x, t, y\rangle \rightarrow |x, t \oplus x, y \oplus f(x)\rangle$$

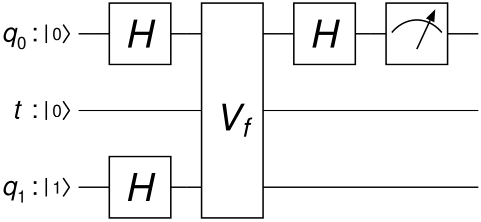
Deutch's algorithm now will not function properly



# Entanglement problems

Suppose the internal temporary computation, $|t\rangle$ in the Deutch example is made explicit in a transformation $V_f$ such that

$$V_f : |x, t, y\rangle \rightarrow |x, t \oplus x, y \oplus f(x)\rangle$$



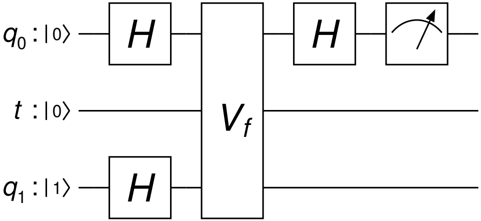

Deutch's algorithm now will not function properly

$$q_0 : |0\rangle \rightarrow |+\rangle = \frac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big), \qquad |t\rangle : |0\rangle, \qquad q_1 : |1\rangle \rightarrow |-\rangle = \frac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big)$$

# Entanglement problems

Suppose the internal temporary computation, $|t\rangle$ in the Deutch example is made explicit in a transformation $V_f$ such that

$$V_f : |x, t, y\rangle \to |x, t \oplus x, y \oplus f(x)\rangle$$




Deutch's algorithm now will not function properly

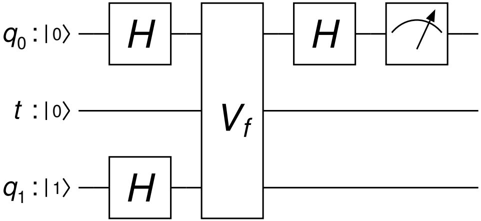
$$q_0 : |0\rangle \to |+\rangle = \frac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big), \quad |t\rangle : |0\rangle, \quad q_1 : |1\rangle \to |-\rangle = \frac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big)$$

$$V_f(|+\rangle|0\rangle|-\rangle) = V_f\left(\frac{1}{\sqrt{2}}\sum_{x=0}^{1}|x\rangle|0\rangle|-\rangle\right)$$

# Entanglement problems

Suppose the internal temporary computation, $|t\rangle$ in the Deutch example is made explicit in a transformation $V_f$ such that

$$V_f : |x, t, y\rangle \rightarrow |x, t \oplus x, y \oplus f(x)\rangle$$



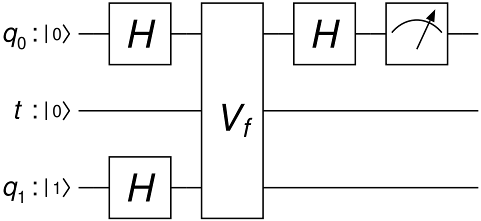

Deutch's algorithm now will not function properly

$$q_0 : |0\rangle \rightarrow |+\rangle = \tfrac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big), \qquad |t\rangle : |0\rangle, \qquad q_1 : |1\rangle \rightarrow |-\rangle = \tfrac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big)$$

$$V_f(|+\rangle|0\rangle|-\rangle) = V_f\left(\frac{1}{\sqrt{2}}\sum_{x=0}^{1}|x\rangle|0\rangle|-\rangle\right) = \frac{1}{\sqrt{2}}\sum_{x=0}^{1}(-1)^{f(x)}|x\rangle|x\rangle|-\rangle$$

# Entanglement problems

Suppose the internal temporary computation, $|t\rangle$ in the Deutch example is made explicit in a transformation $V_f$ such that




$$V_f : |x, t, y\rangle \rightarrow |x, t \oplus x, y \oplus f(x)\rangle$$

Deutch's algorithm now will not function properly

$$q_0 : |0\rangle \rightarrow |+\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \qquad |t\rangle : |0\rangle, \qquad q_1 : |1\rangle \rightarrow |-\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$

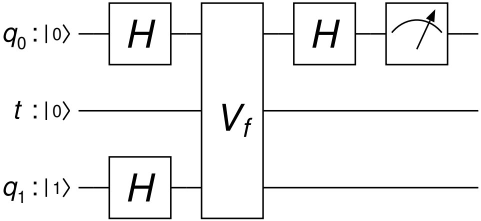
$$V_f(|+\rangle|0\rangle|-\rangle) = V_f\left(\frac{1}{\sqrt{2}}\sum_{x=0}^{1}|x\rangle|0\rangle|-\rangle\right) = \frac{1}{\sqrt{2}}\sum_{x=0}^{1}(-1)^{f(x)}|x\rangle|x\rangle|-\rangle$$

For $f(x)$ constant or balanced the $V_f$ transformation yields

# Entanglement problems

Suppose the internal temporary computation, $|t\rangle$ in the Deutch example is made explicit in a transformation $V_f$ such that

$$V_f : |x, t, y\rangle \rightarrow |x, t \oplus x, y \oplus f(x)\rangle$$




Deutch's algorithm now will not function properly

$$q_0 : |0\rangle \rightarrow |+\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \quad |t\rangle : |0\rangle, \quad q_1 : |1\rangle \rightarrow |-\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$

$$V_f(|+\rangle|0\rangle|-\rangle) = V_f\left(\frac{1}{\sqrt{2}}\sum_{x=0}^{1}|x\rangle|0\rangle|-\rangle\right) = \frac{1}{\sqrt{2}}\sum_{x=0}^{1}(-1)^{f(x)}|x\rangle|x\rangle|-\rangle$$

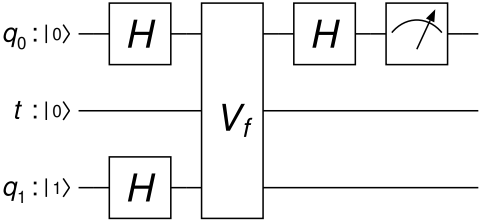
For $f(x)$ constant or balanced the $V_f$ transformation yields

$$f(x) \text{ constant} \quad \longrightarrow \quad V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle + |11\rangle)|-\rangle$$

# Entanglement problems

Suppose the internal temporary computation, $|t\rangle$ in the Deutch example is made explicit in a transformation $V_f$ such that

$$V_f : |x, t, y\rangle \rightarrow |x, t \oplus x, y \oplus f(x)\rangle$$




Deutch's algorithm now will not function properly

$$q_0 : |0\rangle \rightarrow |+\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \quad |t\rangle : |0\rangle, \quad q_1 : |1\rangle \rightarrow |-\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$

$$V_f(|+\rangle|0\rangle|-\rangle) = V_f\left(\frac{1}{\sqrt{2}}\sum_{x=0}^{1}|x\rangle|0\rangle|-\rangle\right) = \frac{1}{\sqrt{2}}\sum_{x=0}^{1}(-1)^{f(x)}|x\rangle|x\rangle|-\rangle$$

For $f(x)$ constant or balanced the $V_f$ transformation yields

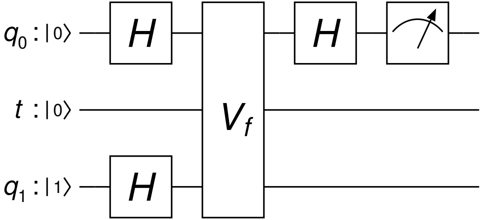
$$f(x) \text{ constant} \longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle + |11\rangle)|-\rangle$$

$$f(x) \text{ balanced} \longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle - |11\rangle)|-\rangle$$

# Entanglement problems

$f(x)$ constant $\longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle + |11\rangle)|-\rangle$

$f(x)$ balanced $\longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle - |11\rangle)|-\rangle$

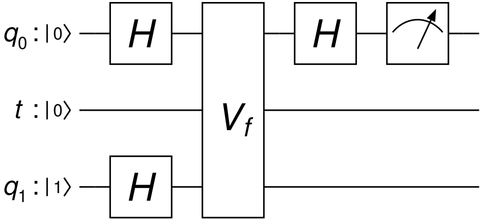


# Entanglement problems

$f(x)$ constant $\longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle + |11\rangle)|-\rangle$

$f(x)$ balanced $\longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle - |11\rangle)|-\rangle$

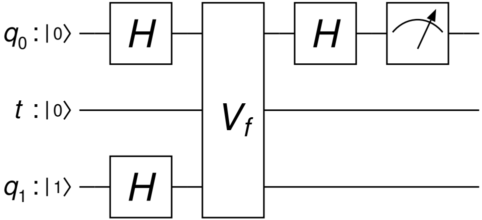
The final step of applying the Hadamard transformation to $q_0$ uses the $H \otimes I \otimes I$ transformation which yields



# Entanglement problems

$f(x)$ constant $\longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle + |11\rangle)|-\rangle$

$f(x)$ balanced $\longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle - |11\rangle)|-\rangle$




The final step of applying the Hadamard transformation to $q_0$ uses the $H \otimes I \otimes I$ transformation which yields

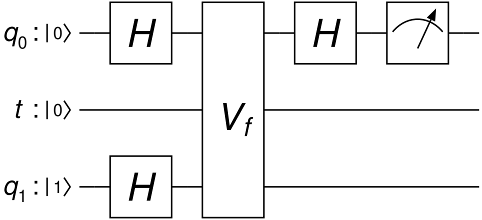
$f(x)$ constant

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

# Entanglement problems

$f(x)$ constant $\longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle + |11\rangle)|-\rangle$

$f(x)$ balanced $\longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle - |11\rangle)|-\rangle$




The final step of applying the Hadamard transformation to $q_0$ uses the $H \otimes I \otimes I$ transformation which yields

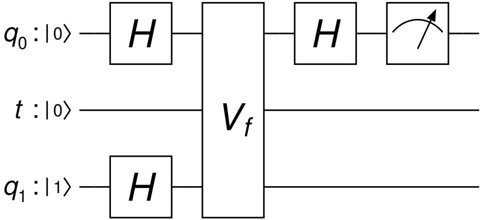
$f(x)$ constant
$$H \otimes I \otimes I(|00\rangle + |11\rangle)|-\rangle = \tfrac{1}{2}(|00\rangle + |10\rangle + |01\rangle - |11\rangle)|-\rangle$$

$f(x)$ balanced
$$H \otimes I \otimes I(|00\rangle - |11\rangle)|-\rangle = \tfrac{1}{2}(|00\rangle + |10\rangle - |01\rangle + |11\rangle)|-\rangle$$

# Entanglement problems

$f(x)$ constant $\longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle + |11\rangle)|-\rangle$

$f(x)$ balanced $\longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle - |11\rangle)|-\rangle$




The final step of applying the Hadamard transformation to $q_0$ uses the $H \otimes I \otimes I$ transformation which yields

$f(x)$ constant $\quad H \otimes I \otimes I(|00\rangle + |11\rangle)|-\rangle = \frac{1}{2}(|00\rangle + |10\rangle + |01\rangle - |11\rangle)|-\rangle$

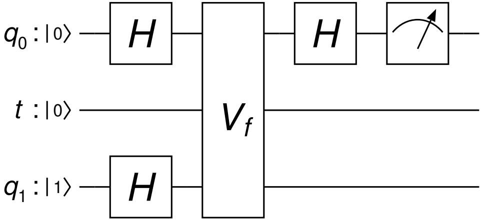
$f(x)$ balanced $\quad H \otimes I \otimes I(|00\rangle - |11\rangle)|-\rangle = \frac{1}{2}(|00\rangle + |10\rangle - |01\rangle + |11\rangle)|-\rangle$

The $q_0$ qubit can be measured to be $|0\rangle$ or $|1\rangle$ with equal probability in both cases since two of the terms do not cancel

# Entanglement problems

$f(x)$ constant $\longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle + |11\rangle)|-\rangle$

$f(x)$ balanced $\longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle - |11\rangle)|-\rangle$




The final step of applying the Hadamard transformation to $q_0$ uses the $H \otimes I \otimes I$ transformation which yields

$f(x)$ constant $\quad H \otimes I \otimes I(|00\rangle + |11\rangle)|-\rangle = \frac{1}{2}(|00\rangle + |10\rangle + |01\rangle - |11\rangle)|-\rangle$

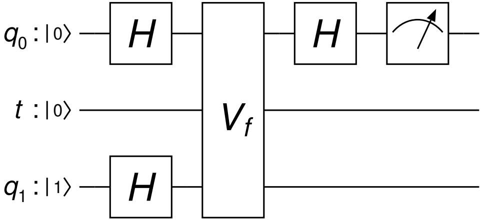
$f(x)$ balanced $\quad H \otimes I \otimes I(|00\rangle - |11\rangle)|-\rangle = \frac{1}{2}(|00\rangle + |10\rangle - |01\rangle + |11\rangle)|-\rangle$

The $q_0$ qubit can be measured to be $|0\rangle$ or $|1\rangle$ with equal probability in both cases since two of the terms do not cancel but by uncomputing the $t$ qubit, the states would be left as

# Entanglement problems

$f(x)$ constant $\longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle + |11\rangle)|-\rangle$

$f(x)$ balanced $\longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle - |11\rangle)|-\rangle$




The final step of applying the Hadamard transformation to $q_0$ uses the $H \otimes I \otimes I$ transformation which yields

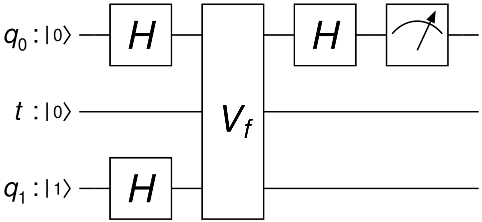
$f(x)$ constant $\quad H \otimes I \otimes I(|00\rangle + |11\rangle)|-\rangle = \frac{1}{2}(|00\rangle + |10\rangle + |01\rangle - |11\rangle)|-\rangle$

$f(x)$ balanced $\quad H \otimes I \otimes I(|00\rangle - |11\rangle)|-\rangle = \frac{1}{2}(|00\rangle + |10\rangle - |01\rangle + |11\rangle)|-\rangle$

The $q_0$ qubit can be measured to be $|0\rangle$ or $|1\rangle$ with equal probability in both cases since two of the terms do not cancel but by uncomputing the $t$ qubit, the states would be left as

$f(x)$ constant $\quad H \otimes I \otimes I(|00\rangle + |10\rangle)|-\rangle = \frac{1}{2}(|00\rangle + |10\rangle + |00\rangle - |10\rangle)|-\rangle$

# Entanglement problems




$f(x)$ constant $\longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle + |11\rangle)|-\rangle$

$f(x)$ balanced $\longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle - |11\rangle)|-\rangle$

The final step of applying the Hadamard transformation to $q_0$ uses the $H \otimes I \otimes I$ transformation which yields

$f(x)$ constant $$H \otimes I \otimes I(|00\rangle + |11\rangle)|-\rangle = \frac{1}{2}(|00\rangle + |10\rangle + |01\rangle - |11\rangle)|-\rangle$$

$f(x)$ balanced $$H \otimes I \otimes I(|00\rangle - |11\rangle)|-\rangle = \frac{1}{2}(|00\rangle + |10\rangle - |01\rangle + |11\rangle)|-\rangle$$

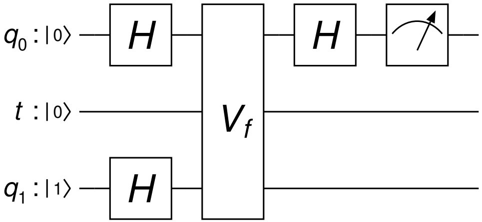
The $q_0$ qubit can be measured to be $|0\rangle$ or $|1\rangle$ with equal probability in both cases since two of the terms do not cancel but by uncomputing the $t$ qubit, the states would be left as

$f(x)$ constant $$H \otimes I \otimes I(|00\rangle + |10\rangle)|-\rangle = \frac{1}{2}(|00\rangle + \cancel{|10\rangle} + |00\rangle - \cancel{|10\rangle})|-\rangle = |0\rangle|0\rangle|-\rangle$$

# Entanglement problems

$f(x)$ constant $\longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle + |11\rangle)|-\rangle$

$f(x)$ balanced $\longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle - |11\rangle)|-\rangle$




The final step of applying the Hadamard transformation to $q_0$ uses the $H \otimes I \otimes I$ transformation which yields

$f(x)$ constant $$H \otimes I \otimes I(|00\rangle + |11\rangle)|-\rangle = \frac{1}{2}(|00\rangle + |10\rangle + |01\rangle - |11\rangle)|-\rangle$$

$f(x)$ balanced $$H \otimes I \otimes I(|00\rangle - |11\rangle)|-\rangle = \frac{1}{2}(|00\rangle + |10\rangle - |01\rangle + |11\rangle)|-\rangle$$

The $q_0$ qubit can be measured to be $|0\rangle$ or $|1\rangle$ with equal probability in both cases since two of the terms do not cancel but by uncomputing the $t$ qubit, the states would be left as

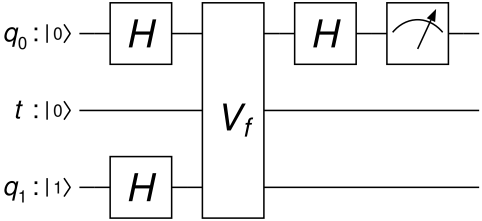
$f(x)$ constant $$H \otimes I \otimes I(|00\rangle + |10\rangle)|-\rangle = \frac{1}{2}(|00\rangle + \cancel{|10\rangle} + |00\rangle - \cancel{|10\rangle})|-\rangle = |0\rangle|0\rangle|-\rangle$$

$f(x)$ balanced $$H \otimes I \otimes I(|00\rangle - |10\rangle)|-\rangle = \frac{1}{2}(|00\rangle + |10\rangle - |00\rangle + |10\rangle)|-\rangle$$

# Entanglement problems

$f(x)$ constant $\longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle + |11\rangle)|-\rangle$

$f(x)$ balanced $\longrightarrow V_f(|+\rangle|0\rangle|-\rangle) = (|00\rangle - |11\rangle)|-\rangle$




The final step of applying the Hadamard transformation to $q_0$ uses the $H \otimes I \otimes I$ transformation which yields

$f(x)$ constant $\quad H \otimes I \otimes I(|00\rangle + |11\rangle)|-\rangle = \frac{1}{2}(|00\rangle + |10\rangle + |01\rangle - |11\rangle)|-\rangle$

$f(x)$ balanced $\quad H \otimes I \otimes I(|00\rangle - |11\rangle)|-\rangle = \frac{1}{2}(|00\rangle + |10\rangle - |01\rangle + |11\rangle)|-\rangle$

The $q_0$ qubit can be measured to be $|0\rangle$ or $|1\rangle$ with equal probability in both cases since two of the terms do not cancel but by uncomputing the $t$ qubit, the states would be left as

$f(x)$ constant $\quad H \otimes I \otimes I(|00\rangle + |10\rangle)|-\rangle = \frac{1}{2}(|00\rangle + \cancel{|10\rangle} + |00\rangle - \cancel{|10\rangle})|-\rangle = |0\rangle|0\rangle|-\rangle$

$f(x)$ balanced $\quad H \otimes I \otimes I(|00\rangle - |10\rangle)|-\rangle = \frac{1}{2}(\cancel{|00\rangle} + |10\rangle - \cancel{|00\rangle} + |10\rangle)|-\rangle = |1\rangle|0\rangle|-\rangle$

# Phase change for a subspace

Suppose we have a superposition state given by $|\psi\rangle = \sum a_i |i\rangle$

# Phase change for a subspace

Suppose we have a superposition state given by $|\psi\rangle = \sum a_i |i\rangle$

We wish to change the phase of every term $|i\rangle$ in the superposition if $|i\rangle \in X$ where $X$ is a subset of the entire space $\{0, 1, \ldots, N-1\}$

# Phase change for a subspace

Suppose we have a superposition state given by $|\psi\rangle = \sum a_i |i\rangle$

We wish to change the phase of every term $|i\rangle$ in the superposition if $|i\rangle \in X$ where $X$ is a subset of the entire space $\{0, 1, \ldots, N-1\}$

The goal is to find an efficient implementation of the transformation $S_X^\phi$ where

# Phase change for a subspace

Suppose we have a superposition state given by $|\psi\rangle = \sum a_i|i\rangle$

We wish to change the phase of every term $|i\rangle$ in the superposition if $|i\rangle \in X$ where $X$ is a subset of the entire space $\{0, 1, \ldots, N-1\}$

The goal is to find an efficient implementation of the transformation $S_X^\phi$ where

$$S_X^\phi \sum_{x=0}^{N-1} a_x|x\rangle$$

# Phase change for a subspace

Suppose we have a superposition state given by $|\psi\rangle = \sum a_i|i\rangle$

We wish to change the phase of every term $|i\rangle$ in the superposition if $|i\rangle \in X$ where $X$ is a subset of the entire space $\{0, 1, \dots, N-1\}$

The goal is to find an efficient implementation of the transformation $S_X^\phi$ where

$$S_X^\phi \sum_{x=0}^{N-1} a_x|x\rangle = \sum_{x\in X} a_x e^{i\phi}|x\rangle + \sum_{x\notin X} a_x|x\rangle$$

# Phase change for a subspace

Suppose we have a superposition state given by $|\psi\rangle = \sum a_i |i\rangle$

We wish to change the phase of every term $|i\rangle$ in the superposition if $|i\rangle \in X$ where $X$ is a subset of the entire space $\{0, 1, \ldots, N-1\}$

The goal is to find an efficient implementation of the transformation $S_X^\phi$ where

$$S_X^\phi \sum_{x=0}^{N-1} a_x |x\rangle = \sum_{x \in X} a_x e^{i\phi} |x\rangle + \sum_{x \notin X} a_x |x\rangle$$

Clearly it is possible to find a brute force implementation using the methods of Chapter 5, however we want an efficient implementation to determine if a state is in a specific subspace

# Phase change for a subspace

Suppose we have a superposition state given by $|\psi\rangle = \sum a_i |i\rangle$

We wish to change the phase of every term $|i\rangle$ in the superposition if $|i\rangle \in X$ where $X$ is a subset of the entire space $\{0, 1, \ldots, N-1\}$

The goal is to find an efficient implementation of the transformation $S_X^\phi$ where

$$S_X^\phi \sum_{x=0}^{N-1} a_x |x\rangle = \sum_{x \in X} a_x e^{i\phi} |x\rangle + \sum_{x \notin X} a_x |x\rangle$$

Clearly it is possible to find a brute force implementation using the methods of Chapter 5, however we want an efficient implementation to determine if a state is in a specific subspace

We want a function $f(x) : \mathbf{Z}_{2^n} \to \mathbf{Z}_2$ that takes the natural numbers (represented by $\mathbf{Z}$) modulo $2^n$ into the natural numbers modulo 2 such that

# Phase change for a subspace

Suppose we have a superposition state given by $|\psi\rangle = \sum a_i|i\rangle$

We wish to change the phase of every term $|i\rangle$ in the superposition if $|i\rangle \in X$ where $X$ is a subset of the entire space $\{0, 1, \ldots, N-1\}$

The goal is to find an efficient implementation of the transformation $S_X^\phi$ where

$$S_X^\phi \sum_{x=0}^{N-1} a_x|x\rangle = \sum_{x\in X} a_x e^{i\phi}|x\rangle + \sum_{x\notin X} a_x|x\rangle$$

Clearly it is possible to find a brute force implementation using the methods of Chapter 5, however we want an efficient implementation to determine if a state is in a specific subspace

We want a function $f(x) : \mathbf{Z}_{2^n} \to \mathbf{Z}_2$ that takes the natural numbers (represented by $\mathbf{Z}$) modulo $2^n$ into the natural numbers modulo 2 such that

$$f(x) = \begin{cases} 1 & x \in X \\ 0 & x \notin X \end{cases}$$

# Phase change for a subspace

Suppose we have a superposition state given by $|\psi\rangle = \sum a_i|i\rangle$

We wish to change the phase of every term $|i\rangle$ in the superposition if $|i\rangle \in X$ where $X$ is a subset of the entire space $\{0, 1, \ldots, N-1\}$

The goal is to find an efficient implementation of the transformation $S_X^\phi$ where

$$S_X^\phi \sum_{x=0}^{N-1} a_x|x\rangle = \sum_{x\in X} a_x e^{i\phi}|x\rangle + \sum_{x\notin X} a_x|x\rangle$$

Clearly it is possible to find a brute force implementation using the methods of Chapter 5, however we want an efficient implementation to determine if a state is in a specific subspace

We want a function $f(x) : \mathbf{Z}_{2^n} \to \mathbf{Z}_2$ that takes the natural numbers (represented by $\mathbf{Z}$) modulo $2^n$ into the natural numbers modulo 2 such that

$$f(x) = \begin{cases} 1 & x \in X \\ 0 & x \notin X \end{cases}$$

The depends on being able to compute membership in $X$ efficiently but if this is possible with a quantum transformation $U_f$ then through the use of a temporary qubit, it is possible to compute $S_X^\phi$

# Phase change for a subspace

The procedure is to use the temporary qubit to compute $f(x)$, then use the result of $f(x)$ to apply the phase change and finally uncompute the temporary qubit to avoid entanglement

# Phase change for a subspace

The procedure is to use the temporary qubit to compute $f(x)$, then use the result of $f(x)$ to apply the phase change and finally uncompute the temporary qubit to avoid entanglement

The pseudo-code (Box 6.2 in text) is as follows

# Phase change for a subspace

The procedure is to use the temporary qubit to compute $f(x)$, then use the result of $f(x)$ to apply the phase change and finally uncompute the temporary qubit to avoid entanglement

The pseudo-code (Box 6.2 in text) is as follows

**define** $Phase_f(\phi)|x[k]\rangle =$

# Phase change for a subspace

The procedure is to use the temporary qubit to compute $f(x)$, then use the result of $f(x)$ to apply the phase change and finally uncompute the temporary qubit to avoid entanglement

The pseudo-code (Box 6.2 in text) is as follows

**define** $Phase_f(\phi)|x[k]\rangle =$

1. **qubit** $a[1]$

# Phase change for a subspace

The procedure is to use the temporary qubit to compute $f(x)$, then use the result of $f(x)$ to apply the phase change and finally uncompute the temporary qubit to avoid entanglement

The pseudo-code (Box 6.2 in text) is as follows

**define** $Phase_f(\phi)|x[k]\rangle =$

1. **qubit** $a[1]$ — create a single qubit $a$ and set it to $|0\rangle$

# Phase change for a subspace

The procedure is to use the temporary qubit to compute $f(x)$, then use the result of $f(x)$ to apply the phase change and finally uncompute the temporary qubit to avoid entanglement

The pseudo-code (Box 6.2 in text) is as follows

**define** $Phase_f(\phi)|x[k]\rangle =$

1. **qubit** $a[1]$ — create a single qubit $a$ and set it to $|0\rangle$
2. $U_f|x, a\rangle$

# Phase change for a subspace

The procedure is to use the temporary qubit to compute $f(x)$, then use the result of $f(x)$ to apply the phase change and finally uncompute the temporary qubit to avoid entanglement

The pseudo-code (Box 6.2 in text) is as follows

**define** $Phase_f(\phi)|x[k]\rangle =$

1. **qubit**$a[1]$ — create a single qubit $a$ and set it to $|0\rangle$
2. $U_f|x, a\rangle$ — compute $f(x) \longrightarrow a$

# Phase change for a subspace

The procedure is to use the temporary qubit to compute $f(x)$, then use the result of $f(x)$ to apply the phase change and finally uncompute the temporary qubit to avoid entanglement

The pseudo-code (Box 6.2 in text) is as follows

**define** $Phase_f(\phi)|x[k]\rangle =$

1. **qubit** $a[1]$ — create a single qubit $a$ and set it to $|0\rangle$
2. $U_f|x, a\rangle$ — compute $f(x) \longrightarrow a$
3. $K(\frac{\phi}{2})|a\rangle$

# Phase change for a subspace

The procedure is to use the temporary qubit to compute $f(x)$, then use the result of $f(x)$ to apply the phase change and finally uncompute the temporary qubit to avoid entanglement

The pseudo-code (Box 6.2 in text) is as follows

**define** $Phase_f(\phi)|x[k]\rangle =$

| | | |
|---|---|---|
| 1. | **qubit** $a[1]$ | create a single qubit $a$ and set it to $\|0\rangle$ |
| 2. | $U_f\|x, a\rangle$ | compute $f(x) \longrightarrow a$ |
| 3. | $K(\frac{\phi}{2})\|a\rangle$ | apply a phase shift |

# Phase change for a subspace

The procedure is to use the temporary qubit to compute $f(x)$, then use the result of $f(x)$ to apply the phase change and finally uncompute the temporary qubit to avoid entanglement

The pseudo-code (Box 6.2 in text) is as follows

**define** $Phase_f(\phi)|x[k]\rangle =$

1. **qubit**$a[1]$ — create a single qubit $a$ and set it to $|0\rangle$
2. $U_f|x, a\rangle$ — compute $f(x) \longrightarrow a$
3. $K(\frac{\phi}{2})|a\rangle$ — apply a phase shift
4. $T(-\frac{\phi}{2})|a\rangle$

# Phase change for a subspace

The procedure is to use the temporary qubit to compute $f(x)$, then use the result of $f(x)$ to apply the phase change and finally uncompute the temporary qubit to avoid entanglement

The pseudo-code (Box 6.2 in text) is as follows

**define** $Phase_f(\phi)|x[k]\rangle =$

| | | |
|---|---|---|
| 1. | **qubit**$a[1]$ | create a single qubit $a$ and set it to $\|0\rangle$ |
| 2. | $U_f\|x, a\rangle$ | compute $f(x) \longrightarrow a$ |
| 3. | $K(\frac{\phi}{2})\|a\rangle$ | apply a phase shift |
| 4. | $T(-\frac{\phi}{2})\|a\rangle$ | apply a phase rotation |

# Phase change for a subspace

The procedure is to use the temporary qubit to compute $f(x)$, then use the result of $f(x)$ to apply the phase change and finally uncompute the temporary qubit to avoid entanglement

The pseudo-code (Box 6.2 in text) is as follows

**define** $Phase_f(\phi)|x[k]\rangle =$

| | | |
|---|---|---|
| 1. | **qubit**$a[1]$ | create a single qubit $a$ and set it to $\|0\rangle$ |
| 2. | $U_f\|x, a\rangle$ | compute $f(x) \longrightarrow a$ |
| 3. | $K(\frac{\phi}{2})\|a\rangle$ | apply a phase shift |
| 4. | $T(-\frac{\phi}{2})\|a\rangle$ | apply a phase rotation |
| 5. | $U_f^{-1}\|x, a\rangle$ | |

# Phase change for a subspace

The procedure is to use the temporary qubit to compute $f(x)$, then use the result of $f(x)$ to apply the phase change and finally uncompute the temporary qubit to avoid entanglement

The pseudo-code (Box 6.2 in text) is as follows

**define** $Phase_f(\phi)|x[k]\rangle =$

| | | |
|---|---|---|
| 1. | **qubit** $a[1]$ | create a single qubit $a$ and set it to $\|0\rangle$ |
| 2. | $U_f\|x,a\rangle$ | compute $f(x) \longrightarrow a$ |
| 3. | $K(\frac{\phi}{2})\|a\rangle$ | apply a phase shift |
| 4. | $T(-\frac{\phi}{2})\|a\rangle$ | apply a phase rotation |
| 5. | $U_f^{-1}\|x,a\rangle$ | uncompute $f(x)$ to disentangle $a$ |

# Phase change for a subspace

The procedure is to use the temporary qubit to compute $f(x)$, then use the result of $f(x)$ to apply the phase change and finally uncompute the temporary qubit to avoid entanglement

The pseudo-code (Box 6.2 in text) is as follows

**define** $Phase_f(\phi)|x[k]\rangle =$

| | | |
|---|---|---|
| 1. | **qubit** $a[1]$ | create a single qubit $a$ and set it to $|0\rangle$ |
| 2. | $U_f|x,a\rangle$ | compute $f(x) \longrightarrow a$ |
| 3. | $K(\frac{\phi}{2})|a\rangle$ | apply a phase shift |
| 4. | $T(-\frac{\phi}{2})|a\rangle$ | apply a phase rotation |
| 5. | $U_f^{-1}|x,a\rangle$ | uncompute $f(x)$ to disentangle $a$ |

The $TK$ sequence serves to apply a rotation only if $a$ is equal to $|1\rangle$

# Phase change for a subspace

The procedure is to use the temporary qubit to compute $f(x)$, then use the result of $f(x)$ to apply the phase change and finally uncompute the temporary qubit to avoid entanglement

The pseudo-code (Box 6.2 in text) is as follows

**define** $Phase_f(\phi)|x[k]\rangle =$

| | | |
|---|---|---|
| 1. | **qubit** $a[1]$ | create a single qubit $a$ and set it to $|0\rangle$ |
| 2. | $U_f|x,a\rangle$ | compute $f(x) \longrightarrow a$ |
| 3. | $K(\frac{\phi}{2})|a\rangle$ | apply a phase shift |
| 4. | $T(-\frac{\phi}{2})|a\rangle$ | apply a phase rotation |
| 5. | $U_f^{-1}|x,a\rangle$ | uncompute $f(x)$ to disentangle $a$ |

The $TK$ sequence serves to apply a rotation only if $a$ is equal to $|1\rangle$

$$T(-\tfrac{\phi}{2})K(\tfrac{\phi}{2})$$

# Phase change for a subspace

The procedure is to use the temporary qubit to compute $f(x)$, then use the result of $f(x)$ to apply the phase change and finally uncompute the temporary qubit to avoid entanglement

The pseudo-code (Box 6.2 in text) is as follows

**define** $Phase_f(\phi)|x[k]\rangle =$

| | | |
|---|---|---|
| 1. | **qubit** $a[1]$ | create a single qubit $a$ and set it to $\|0\rangle$ |
| 2. | $U_f\|x, a\rangle$ | compute $f(x) \longrightarrow a$ |
| 3. | $K(\frac{\phi}{2})\|a\rangle$ | apply a phase shift |
| 4. | $T(-\frac{\phi}{2})\|a\rangle$ | apply a phase rotation |
| 5. | $U_f^{-1}\|x, a\rangle$ | uncompute $f(x)$ to disentangle $a$ |

The $TK$ sequence serves to apply a rotation only if $a$ is equal to $|1\rangle$

$$T(-\tfrac{\phi}{2})K(\tfrac{\phi}{2}) = \begin{pmatrix} e^{-i\phi/2} & 0 \\ 0 & e^{+i\phi/2} \end{pmatrix} \begin{pmatrix} e^{+i\phi/2} & 0 \\ 0 & e^{+i\phi/2} \end{pmatrix}$$

# Phase change for a subspace

The procedure is to use the temporary qubit to compute $f(x)$, then use the result of $f(x)$ to apply the phase change and finally uncompute the temporary qubit to avoid entanglement

The pseudo-code (Box 6.2 in text) is as follows

**define** $Phase_f(\phi)|x[k]\rangle =$

| | | |
|---|---|---|
| 1. | **qubit** $a[1]$ | create a single qubit $a$ and set it to $\|0\rangle$ |
| 2. | $U_f\|x,a\rangle$ | compute $f(x) \longrightarrow a$ |
| 3. | $K(\frac{\phi}{2})\|a\rangle$ | apply a phase shift |
| 4. | $T(-\frac{\phi}{2})\|a\rangle$ | apply a phase rotation |
| 5. | $U_f^{-1}\|x,a\rangle$ | uncompute $f(x)$ to disentangle $a$ |

The $TK$ sequence serves to apply a rotation only if $a$ is equal to $|1\rangle$

$$T(-\tfrac{\phi}{2})K(\tfrac{\phi}{2}) = \begin{pmatrix} e^{-i\phi/2} & 0 \\ 0 & e^{+i\phi/2} \end{pmatrix} \begin{pmatrix} e^{+i\phi/2} & 0 \\ 0 & e^{+i\phi/2} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & e^{+i\phi} \end{pmatrix}$$

# Phase change of $\pi$

For the special case of $\phi = \pi$ there is an even simpler implementation

# Phase change of $\pi$

For the special case of $\phi = \pi$ there is an even simpler implementation

$S_X^\pi$ can be implemented by starting with the temporary qubit $b$ in a superposition state $b : |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$

# Phase change of $\pi$

For the special case of $\phi = \pi$ there is an even simpler implementation

$S_X^\pi$ can be implemented by starting with the temporary qubit $b$ in a superposition state $b: |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$

Suppose the initial state is given by

# Phase change of $\pi$

For the special case of $\phi = \pi$ there is an even simpler implementation

$S_X^\pi$ can be implemented by starting with the temporary qubit $b$ in a superposition state $b : |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$

Suppose the initial state is given by

$$|\psi\rangle = \sum_{x \in X} a_x |x\rangle + \sum_{x \notin X} a_x |x\rangle$$

# Phase change of $\pi$

For the special case of $\phi = \pi$ there is an even simpler implementation

$S_X^\pi$ can be implemented by starting with the temporary qubit $b$ in a superposition state $b : |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$

Suppose the initial state is given by

$$|\psi\rangle = \sum_{x \in X} a_x |x\rangle + \sum_{x \notin X} a_x |x\rangle$$

$$U_f(|\psi\rangle \otimes |-\rangle) = U_f\left(\sum_{x \in X} a_x |x\rangle \otimes |-\rangle\right) + U_f\left(\sum_{x \notin X} a_x |x\rangle \otimes |-\rangle\right)$$

# Phase change of $\pi$

For the special case of $\phi = \pi$ there is an even simpler implementation

$S_X^\pi$ can be implemented by starting with the temporary qubit $b$ in a superposition state $b : |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$

Suppose the initial state is given by

$$|\psi\rangle = \sum_{x\in X} a_x|x\rangle + \sum_{x\notin X} a_x|x\rangle$$

$$U_f(|\psi\rangle \otimes |-\rangle) = U_f\left(\sum_{x\in X} a_x|x\rangle \otimes |-\rangle\right) + U_f\left(\sum_{x\notin X} a_x|x\rangle \otimes |-\rangle\right)$$

$$= -\left(\sum_{x\in X} a_x|x\rangle \otimes |-\rangle\right) + \left(\sum_{x\notin X} a_x|x\rangle \otimes |-\rangle\right)$$

# Phase change of $\pi$

For the special case of $\phi = \pi$ there is an even simpler implementation

$S_X^\pi$ can be implemented by starting with the temporary qubit $b$ in a superposition state $b : |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$

Suppose the initial state is given by

$$|\psi\rangle = \sum_{x\in X} a_x|x\rangle + \sum_{x\notin X} a_x|x\rangle$$

$$U_f(|\psi\rangle \otimes |-\rangle) = U_f\left(\sum_{x\in X} a_x|x\rangle \otimes |-\rangle\right) + U_f\left(\sum_{x\notin X} a_x|x\rangle \otimes |-\rangle\right)$$

$$= -\left(\sum_{x\in X} a_x|x\rangle \otimes |-\rangle\right) + \left(\sum_{x\notin X} a_x|x\rangle \otimes |-\rangle\right) = (S_X^\pi|\psi\rangle) \otimes |-\rangle$$

# Phase change of $\pi$

For the special case of $\phi = \pi$ there is an even simpler implementation

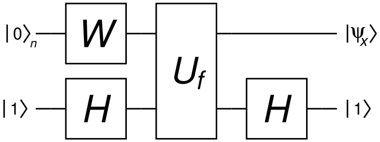
$S_X^{\pi}$ can be implemented by starting with the temporary qubit $b$ in a superposition state $b : |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$

Suppose the initial state is given by

$$|\psi\rangle = \sum_{x \in X} a_x |x\rangle + \sum_{x \notin X} a_x |x\rangle$$

$$U_f(|\psi\rangle \otimes |-\rangle) = U_f\left(\sum_{x \in X} a_x |x\rangle \otimes |-\rangle\right) + U_f\left(\sum_{x \notin X} a_x |x\rangle \otimes |-\rangle\right)$$

$$= -\left(\sum_{x \in X} a_x |x\rangle \otimes |-\rangle\right) + \left(\sum_{x \notin X} a_x |x\rangle \otimes |-\rangle\right) = (S_X^{\pi}|\psi\rangle) \otimes |-\rangle$$



# Phase change of $\pi$

For the special case of $\phi = \pi$ there is an even simpler implementation

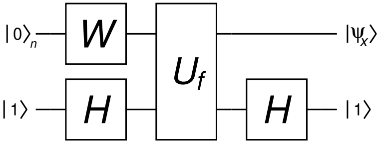
$S_X^\pi$ can be implemented by starting with the temporary qubit $b$ in a superposition state $b : |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$

Suppose the initial state is given by

$$|\psi\rangle = \sum_{x \in X} a_x |x\rangle + \sum_{x \notin X} a_x |x\rangle$$

$$U_f(|\psi\rangle \otimes |-\rangle) = U_f\left(\sum_{x \in X} a_x|x\rangle \otimes |-\rangle\right) + U_f\left(\sum_{x \notin X} a_x|x\rangle \otimes |-\rangle\right)$$

$$= -\left(\sum_{x \in X} a_x|x\rangle \otimes |-\rangle\right) + \left(\sum_{x \notin X} a_x|x\rangle \otimes |-\rangle\right) = (S_X^\pi|\psi\rangle) \otimes |-\rangle$$




The circuit starts with a uniform superposition of an $n$-qubit register and an acilla qubit in the $|1\rangle$ state to create the superposition $|\psi_X\rangle = \sum(-1)^{f(x)}|x\rangle$

# State-dependent phase changes

Suppose we wish to apply a phase shift that depends on the state of a specific qubit, $|x\rangle \to e^{i\phi(x)}|x\rangle$ where there is an associated function $f : \mathbf{Z_n} \to \mathbf{Z_s}$ that is efficiently computable

# State-dependent phase changes

Suppose we wish to apply a phase shift that depends on the state of a specific qubit, $|x\rangle \to e^{i\phi(x)}|x\rangle$ where there is an associated function $f : \mathbf{Z_n} \to \mathbf{Z_s}$ that is efficiently computable

The $i^{th}$ bit of $f(x)$ is the $i^{th}$ term of the binary expansion for the phase, $\phi(x) \approx 2\pi f(x)/2^s$

# State-dependent phase changes

Suppose we wish to apply a phase shift that depends on the state of a specific qubit, $|x\rangle \to e^{i\phi(x)}|x\rangle$ where there is an associated function $f : \mathbf{Z_n} \to \mathbf{Z_s}$ that is efficiently computable

The $i^{th}$ bit of $f(x)$ is the $i^{th}$ term of the binary expansion for the phase, $\phi(x) \approx 2\pi f(x)/2^s$

Given a transformation $U_f$ that is efficient, it is possible to perform the state-dependent phase shift in $O(s)$ steps plus 2 invocations of $U_f$

# State-dependent phase changes

Suppose we wish to apply a phase shift that depends on the state of a specific qubit, $|x\rangle \rightarrow e^{i\phi(x)}|x\rangle$ where there is an associated function $f : \mathbf{Z_n} \rightarrow \mathbf{Z_s}$ that is efficiently computable

The $i^{th}$ bit of $f(x)$ is the $i^{th}$ term of the binary expansion for the phase, $\phi(x) \approx 2\pi f(x)/2^s$

Given a transformation $U_f$ that is efficient, it is possible to perform the state-dependent phase shift in $O(s)$ steps plus 2 invocations of $U_f$

Suppose that $f(x) = x$, we want a subroutine that changes the phase of an s-qubit standard basis state $|x\rangle$ by $\phi(x) = 2\pi x/2^s$ using the transformation

# State-dependent phase changes

Suppose we wish to apply a phase shift that depends on the state of a specific qubit, $|x\rangle \to e^{i\phi(x)}|x\rangle$ where there is an associated function $f : \mathbf{Z_n} \to \mathbf{Z_s}$ that is efficiently computable

The $i^{th}$ bit of $f(x)$ is the $i^{th}$ term of the binary expansion for the phase, $\phi(x) \approx 2\pi f(x)/2^s$

Given a transformation $U_f$ that is efficient, it is possible to perform the state-dependent phase shift in $O(s)$ steps plus 2 invocations of $U_f$

Suppose that $f(x) = x$, we want a subroutine that changes the phase of an s-qubit standard basis state $|x\rangle$ by $\phi(x) = 2\pi x/2^s$ using the transformation

$$P(\phi) = T\left(-\frac{\phi}{2}\right) K\left(\frac{\phi}{2}\right)$$

# State-dependent phase changes

Suppose we wish to apply a phase shift that depends on the state of a specific qubit, $|x\rangle \rightarrow e^{i\phi(x)}|x\rangle$ where there is an associated function $f : \mathbf{Z_n} \rightarrow \mathbf{Z_s}$ that is efficiently computable

The $i^{th}$ bit of $f(x)$ is the $i^{th}$ term of the binary expansion for the phase, $\phi(x) \approx 2\pi f(x)/2^s$

Given a transformation $U_f$ that is efficient, it is possible to perform the state-dependent phase shift in $O(s)$ steps plus 2 invocations of $U_f$

Suppose that $f(x) = x$, we want a subroutine that changes the phase of an s-qubit standard basis state $|x\rangle$ by $\phi(x) = 2\pi x/2^s$ using the transformation

$$P(\phi) = T\left(-\frac{\phi}{2}\right) K\left(\frac{\phi}{2}\right) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}$$

# State-dependent phase changes

Suppose we wish to apply a phase shift that depends on the state of a specific qubit, $|x\rangle \to e^{i\phi(x)}|x\rangle$ where there is an associated function $f : \mathbf{Z_n} \to \mathbf{Z_s}$ that is efficiently computable

The $i^{th}$ bit of $f(x)$ is the $i^{th}$ term of the binary expansion for the phase, $\phi(x) \approx 2\pi f(x)/2^s$

Given a transformation $U_f$ that is efficient, it is possible to perform the state-dependent phase shift in $O(s)$ steps plus 2 invocations of $U_f$

Suppose that $f(x) = x$, we want a subroutine that changes the phase of an s-qubit standard basis state $|x\rangle$ by $\phi(x) = 2\pi x/2^s$ using the transformation

$$P(\phi) = T\left(-\frac{\phi}{2}\right) K\left(\frac{\phi}{2}\right) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}$$

$$\textbf{define } \mathit{Phase}\ |a[s]\rangle =$$

# State-dependent phase changes

Suppose we wish to apply a phase shift that depends on the state of a specific qubit, $|x\rangle \to e^{i\phi(x)}|x\rangle$ where there is an associated function $f : \mathbf{Z_n} \to \mathbf{Z_s}$ that is efficiently computable

The $i^{th}$ bit of $f(x)$ is the $i^{th}$ term of the binary expansion for the phase, $\phi(x) \approx 2\pi f(x)/2^s$

Given a transformation $U_f$ that is efficient, it is possible to perform the state-dependent phase shift in $O(s)$ steps plus 2 invocations of $U_f$

Suppose that $f(x) = x$, we want a subroutine that changes the phase of an s-qubit standard basis state $|x\rangle$ by $\phi(x) = 2\pi x/2^s$ using the transformation

$$P(\phi) = T\left(-\frac{\phi}{2}\right) K\left(\frac{\phi}{2}\right) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}$$

**define** *Phase* $|a[s]\rangle =$

1. **for** $i \in [0 \dots s-1]$ — loop over all $s$ bits in register $|a\rangle$

# State-dependent phase changes

Suppose we wish to apply a phase shift that depends on the state of a specific qubit, $|x\rangle \to e^{i\phi(x)}|x\rangle$ where there is an associated function $f : \mathbf{Z_n} \to \mathbf{Z_s}$ that is efficiently computable

The $i^{th}$ bit of $f(x)$ is the $i^{th}$ term of the binary expansion for the phase, $\phi(x) \approx 2\pi f(x)/2^s$

Given a transformation $U_f$ that is efficient, it is possible to perform the state-dependent phase shift in $O(s)$ steps plus 2 invocations of $U_f$

Suppose that $f(x) = x$, we want a subroutine that changes the phase of an s-qubit standard basis state $|x\rangle$ by $\phi(x) = 2\pi x/2^s$ using the transformation

$$P(\phi) = T\left(-\frac{\phi}{2}\right) K\left(\frac{\phi}{2}\right) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}$$

**define** $Phase\ |a[s]\rangle =$

1. **for** $i \in [0 \ldots s-1]$ — loop over all $s$ bits in register $|a\rangle$
2. $P\left(\frac{2\pi}{2^i}\right)|a_i\rangle$ — apply the specified rotation to the $i^{th}$ qubit

# State-dependent phase changes

Using the subroutine $Phase\ : |a\rangle \to e^{i2\pi s/2^s}$ it is now possible to write a program that implements the $n$-qubit transformation $Phase_f\ : |x\rangle \to e^{i2\pi f(x)/2^s}$

# State-dependent phase changes

Using the subroutine *Phase* $: |a\rangle \rightarrow e^{i2\pi s/2^s}$ it is now possible to write a program that implements the $n$-qubit transformation $Phase_f : |x\rangle \rightarrow e^{i2\pi f(x)/2^s}$

$$\textbf{define } Phase_f \ |x[k]\rangle =$$

# State-dependent phase changes

Using the subroutine $Phase\ : |a\rangle \rightarrow e^{i2\pi s/2^s}$ it is now possible to write a program that implements the $n$-qubit transformation $Phase_f\ : |x\rangle \rightarrow e^{i2\pi f(x)/2^s}$

**define** $Phase_f\ |x[k]\rangle =$

1. **qubit** $a[s]$ — create an $s$-qubit temporary register

# State-dependent phase changes

Using the subroutine $Phase\ : |a\rangle \rightarrow e^{i2\pi s/2^s}$ it is now possible to write a program that implements the $n$-qubit transformation $Phase_f\ : |x\rangle \rightarrow e^{i2\pi f(x)/2^s}$

| | | |
|---|---|---|
| | **define** $Phase_f\ |x[k]\rangle =$ | |
| 1. | **qubit** $a[s]$ | create an $s$-qubit temporary register |
| 2. | $U_f|x\rangle|a\rangle$ | compute $f$ in $a$ |

# State-dependent phase changes

Using the subroutine $Phase \; : |a\rangle \rightarrow e^{i2\pi s/2^s}$ it is now possible to write a program that implements the $n$-qubit transformation $Phase_f \; : |x\rangle \rightarrow e^{i2\pi f(x)/2^s}$

**define** $Phase_f \; |x[k]\rangle =$

| | | |
|---|---|---|
| 1. | **qubit** $a[s]$ | create an $s$-qubit temporary register |
| 2. | $U_f|x\rangle|a\rangle$ | compute $f$ in $a$ |
| 3. | $Phase \; |a\rangle$ | perform phase shift by $2\pi a/2^s$ |

# State-dependent phase changes

Using the subroutine $Phase\ : |a\rangle \to e^{i2\pi s/2^s}$ it is now possible to write a program that implements the $n$-qubit transformation $Phase_f\ : |x\rangle \to e^{i2\pi f(x)/2^s}$

**define** $Phase_f\ |x[k]\rangle =$

| | | |
|---|---|---|
| 1. | **qubit** $a[s]$ | create an $s$-qubit temporary register |
| 2. | $U_f\|x\rangle\|a\rangle$ | compute $f$ in $a$ |
| 3. | $Phase\ \|a\rangle$ | perform phase shift by $2\pi a/2^s$ |
| 4. | $U_f^{-1}\|x\rangle\|a\rangle$ | uncompute $f$ |

# State-dependent phase changes

Using the subroutine $Phase : |a\rangle \rightarrow e^{i2\pi s/2^s}$ it is now possible to write a program that implements the $n$-qubit transformation $Phase_f : |x\rangle \rightarrow e^{i2\pi f(x)/2^s}$

**define** $Phase_f\ |x[k]\rangle =$

| | | |
|---|---|---|
| 1. | **qubit** $a[s]$ | create an $s$-qubit temporary register |
| 2. | $U_f|x\rangle|a\rangle$ | compute $f$ in $a$ |
| 3. | $Phase\ |a\rangle$ | perform phase shift by $2\pi a/2^s$ |
| 4. | $U_f^{-1}|x\rangle|a\rangle$ | uncompute $f$ |

Step 2 entangles $|a\rangle$ with $|x\rangle$ and is set to the binary expansion of $\phi(x)$ for the desired phase shifts to $|x\rangle$

# State-dependent phase changes

Using the subroutine $Phase : |a\rangle \rightarrow e^{i2\pi s/2^s}$ it is now possible to write a program that implements the $n$-qubit transformation $Phase_f : |x\rangle \rightarrow e^{i2\pi f(x)/2^s}$

**define** $Phase_f \; |x[k]\rangle =$

| | | |
|---|---|---|
| 1. | **qubit** $a[s]$ | create an $s$-qubit temporary register |
| 2. | $U_f|x\rangle|a\rangle$ | compute $f$ in $a$ |
| 3. | $Phase \; |a\rangle$ | perform phase shift by $2\pi a/2^s$ |
| 4. | $U_f^{-1}|x\rangle|a\rangle$ | uncompute $f$ |

Step 2 entangles $|a\rangle$ with $|x\rangle$ and is set to the binary expansion of $\phi(x)$ for the desired phase shifts to $|x\rangle$

Step 3 changes the phase of $|a\rangle$ and also of $|x\rangle$ because they are entangled

# State-dependent phase changes

Using the subroutine $Phase\ : |a\rangle \rightarrow e^{i2\pi s/2^s}$ it is now possible to write a program that implements the $n$-qubit transformation $Phase_f\ : |x\rangle \rightarrow e^{i2\pi f(x)/2^s}$

**define** $Phase_f\ |x[k]\rangle =$

| | | |
|---|---|---|
| 1. | **qubit** $a[s]$ | create an $s$-qubit temporary register |
| 2. | $U_f|x\rangle|a\rangle$ | compute $f$ in $a$ |
| 3. | $Phase\ |a\rangle$ | perform phase shift by $2\pi a/2^s$ |
| 4. | $U_f^{-1}|x\rangle|a\rangle$ | uncompute $f$ |

Step 2 entangles $|a\rangle$ with $|x\rangle$ and is set to the binary expansion of $\phi(x)$ for the desired phase shifts to $|x\rangle$

Step 3 changes the phase of $|a\rangle$ and also of $|x\rangle$ because they are entangled

Step 4 unentangles $|a\rangle$ from $|x\rangle$ leaving it in the desired state

# State-dependent amplitude shifts

We wish to rotate each term in a superposition by a single qubit rotation $R(\beta(x))$ where $\beta(x)$ is state-dependent such that $|x\rangle \otimes |b\rangle \to |x\rangle \otimes (R(\beta(x))|b\rangle$

# State-dependent amplitude shifts

We wish to rotate each term in a superposition by a single qubit rotation $R(\beta(x))$ where $\beta(x)$ is state-dependent such that $|x\rangle \otimes |b\rangle \to |x\rangle \otimes (R(\beta(x))|b\rangle$

If $\beta(x) \approx 2\pi f(x)/2^s$ and $f : \mathbf{Z}_n \to \mathbf{Z}_s$ define a subroutine

# State-dependent amplitude shifts

We wish to rotate each term in a superposition by a single qubit rotation $R(\beta(x))$ where $\beta(x)$ is state-dependent such that $|x\rangle \otimes |b\rangle \rightarrow |x\rangle \otimes (R(\beta(x))|b\rangle$

If $\beta(x) \approx 2\pi f(x)/2^s$ and $f : \mathbf{Z}_n \rightarrow \mathbf{Z}_s$ define a subroutine

$$\textbf{define } Rot \ |a[s]\rangle|b[1]\rangle =$$

# State-dependent amplitude shifts

We wish to rotate each term in a superposition by a single qubit rotation $R(\beta(x))$ where $\beta(x)$ is state-dependent such that $|x\rangle \otimes |b\rangle \rightarrow |x\rangle \otimes (R(\beta(x))|b\rangle$

If $\beta(x) \approx 2\pi f(x)/2^s$ and $f : \mathbf{Z}_n \rightarrow \mathbf{Z}_s$ define a subroutine

**define** $Rot\ |a[s]\rangle|b[1]\rangle =$

1. **for** $i \in [0 \ldots s-1]$ loop over all $s$ bits in register $|a\rangle$

# State-dependent amplitude shifts

We wish to rotate each term in a superposition by a single qubit rotation $R(\beta(x))$ where $\beta(x)$ is state-dependent such that $|x\rangle \otimes |b\rangle \rightarrow |x\rangle \otimes (R(\beta(x))|b\rangle$

If $\beta(x) \approx 2\pi f(x)/2^s$ and $f : \mathbf{Z}_n \rightarrow \mathbf{Z}_s$ define a subroutine

**define** $Rot\ |a[s]\rangle|b[1]\rangle =$

1. **for** $i \in [0 \ldots s-1]$ — loop over all $s$ bits in register $|a\rangle$
2. $|a_i\rangle$ **control**$R\left(\frac{2\pi}{2^i}\right)|b\rangle$ — apply a controlled rotation to the $|b\rangle$ qubit

# State-dependent amplitude shifts

We wish to rotate each term in a superposition by a single qubit rotation $R(\beta(x))$ where $\beta(x)$ is state-dependent such that $|x\rangle \otimes |b\rangle \to |x\rangle \otimes (R(\beta(x))|b\rangle$

If $\beta(x) \approx 2\pi f(x)/2^s$ and $f : \mathbf{Z}_n \to \mathbf{Z}_s$ define a subroutine

**define** $Rot\ |a[s]\rangle|b[1]\rangle =$

| | | |
|---|---|---|
| 1. | **for** $i \in [0 \ldots s-1]$ | loop over all $s$ bits in register $\|a\rangle$ |
| 2. | $\|a_i\rangle$ **control** $R\left(\frac{2\pi}{2^i}\right)\|b\rangle$ | apply a controlled rotation to the $\|b\rangle$ qubit |

The full program is thus

# State-dependent amplitude shifts

We wish to rotate each term in a superposition by a single qubit rotation $R(\beta(x))$ where $\beta(x)$ is state-dependent such that $|x\rangle \otimes |b\rangle \to |x\rangle \otimes (R(\beta(x))|b\rangle$

If $\beta(x) \approx 2\pi f(x)/2^s$ and $f : \mathbf{Z}_n \to \mathbf{Z}_s$ define a subroutine

**define** $Rot\ |a[s]\rangle|b[1]\rangle =$

| | | |
|---|---|---|
| 1. | **for** $i \in [0 \ldots s-1]$ | loop over all $s$ bits in register $\|a\rangle$ |
| 2. | $\|a_i\rangle$ **control** $R\left(\frac{2\pi}{2^i}\right)\|b\rangle$ | apply a controlled rotation to the $\|b\rangle$ qubit |

The full program is thus

**define** $Rot_f\ |x[k]\rangle|b[1]\rangle =$

# State-dependent amplitude shifts

We wish to rotate each term in a superposition by a single qubit rotation $R(\beta(x))$ where $\beta(x)$ is state-dependent such that $|x\rangle \otimes |b\rangle \to |x\rangle \otimes (R(\beta(x))|b\rangle$

If $\beta(x) \approx 2\pi f(x)/2^s$ and $f : \mathbf{Z}_n \to \mathbf{Z}_s$ define a subroutine

**define** $Rot\ |a[s]\rangle|b[1]\rangle =$

1. **for** $i \in [0 \ldots s-1]$ — loop over all $s$ bits in register $|a\rangle$
2. $|a_i\rangle$ **control**$R\left(\frac{2\pi}{2^i}\right)|b\rangle$ — apply a controlled rotation to the $|b\rangle$ qubit

The full program is thus

**define** $Rot_f\ |x[k]\rangle|b[1]\rangle =$

1. **qubit** $a[s]$ — create an $s$-qubit temporary register

# State-dependent amplitude shifts

We wish to rotate each term in a superposition by a single qubit rotation $R(\beta(x))$ where $\beta(x)$ is state-dependent such that $|x\rangle \otimes |b\rangle \rightarrow |x\rangle \otimes (R(\beta(x))|b\rangle$

If $\beta(x) \approx 2\pi f(x)/2^s$ and $f : \mathbf{Z}_n \rightarrow \mathbf{Z}_s$ define a subroutine

**define** $Rot\ |a[s]\rangle|b[1]\rangle =$

| | | |
|---|---|---|
| 1. | **for** $i \in [0 \ldots s-1]$ | loop over all $s$ bits in register $\|a\rangle$ |
| 2. | $\|a_i\rangle$ **control**$R\left(\frac{2\pi}{2^i}\right)\|b\rangle$ | apply a controlled rotation to the $\|b\rangle$ qubit |

The full program is thus

**define** $Rot_f\ |x[k]\rangle|b[1]\rangle =$

| | | |
|---|---|---|
| 1. | **qubit** $a[s]$ | create an $s$-qubit temporary register |
| 2. | $U_f\|x\rangle\|a\rangle$ | compute $f$ in $a$ |

# State-dependent amplitude shifts

We wish to rotate each term in a superposition by a single qubit rotation $R(\beta(x))$ where $\beta(x)$ is state-dependent such that $|x\rangle \otimes |b\rangle \rightarrow |x\rangle \otimes (R(\beta(x))|b\rangle$

If $\beta(x) \approx 2\pi f(x)/2^s$ and $f : \mathbf{Z}_n \rightarrow \mathbf{Z}_s$ define a subroutine

**define** $Rot\ |a[s]\rangle|b[1]\rangle =$

| | | |
|---|---|---|
| 1. | **for** $i \in [0 \ldots s-1]$ | loop over all $s$ bits in register $\|a\rangle$ |
| 2. | $\|a_i\rangle$ **control**$R\left(\frac{2\pi}{2^i}\right)\|b\rangle$ | apply a controlled rotation to the $\|b\rangle$ qubit |

The full program is thus

**define** $Rot_f\ |x[k]\rangle|b[1]\rangle =$

| | | |
|---|---|---|
| 1. | **qubit** $a[s]$ | create an $s$-qubit temporary register |
| 2. | $U_f\|x\rangle\|a\rangle$ | compute $f$ in $a$ |
| 3. | $Rot\ \|a, b\rangle$ | perform rotation by $2\pi a/2^s$ |

# State-dependent amplitude shifts

We wish to rotate each term in a superposition by a single qubit rotation $R(\beta(x))$ where $\beta(x)$ is state-dependent such that $|x\rangle \otimes |b\rangle \rightarrow |x\rangle \otimes (R(\beta(x))|b\rangle$

If $\beta(x) \approx 2\pi f(x)/2^s$ and $f : \mathbf{Z}_n \rightarrow \mathbf{Z}_s$ define a subroutine

**define** $Rot\ |a[s]\rangle|b[1]\rangle =$

| | | |
|---|---|---|
| 1. | **for** $i \in [0 \ldots s-1]$ | loop over all $s$ bits in register $\|a\rangle$ |
| 2. | $\|a_i\rangle$ **control** $R\left(\frac{2\pi}{2^i}\right)\|b\rangle$ | apply a controlled rotation to the $\|b\rangle$ qubit |

The full program is thus

**define** $Rot_f\ |x[k]\rangle|b[1]\rangle =$

| | | |
|---|---|---|
| 1. | **qubit** $a[s]$ | create an $s$-qubit temporary register |
| 2. | $U_f\|x\rangle\|a\rangle$ | compute $f$ in $a$ |
| 3. | $Rot\ \|a, b\rangle$ | perform rotation by $2\pi a/2^s$ |
| 4. | $U_f^{-1}\|x\rangle\|a\rangle$ | uncompute $f$ |

# State-dependent amplitude shifts

**define** $Rot_f\ |x[k]\rangle|b[1]\rangle =$

| | | |
|---|---|---|
| 1. | **qubit** $a[s]$ | create an $s$-qubit temporary register |
| 2. | $U_f|x\rangle|a\rangle$ | compute $f$ in $a$ |
| 3. | $Rot\ |a, b\rangle$ | perform rotation by $2\pi a/2^s$ |
| 4. | $U_f^{-1}|x\rangle|a\rangle$ | uncompute $f$ |

# State-dependent amplitude shifts

**define** $Rot_f \; |x[k]\rangle|b[1]\rangle =$

1. **qubit** $a[s]$ — create an $s$-qubit temporary register
2. $U_f|x\rangle|a\rangle$ — compute $f$ in $a$
3. $Rot \; |a, b\rangle$ — perform rotation by $2\pi a/2^s$
4. $U_f^{-1}|x\rangle|a\rangle$ — uncompute $f$