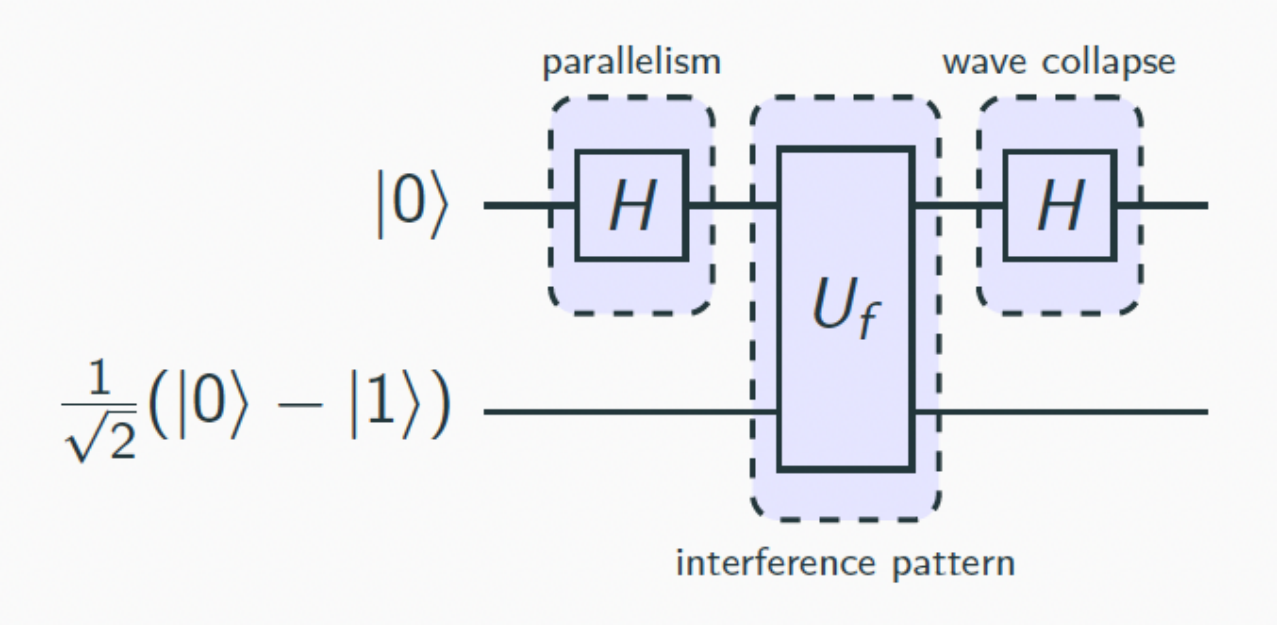
Lecture 7
Query Algorithms and Phase Kick-back

Luís Soares Barbosa
www.di.uminho.pt/~lsb/

Fundamentos de Computação Quântica
Universidade do Minho
2025-2026

Revisiting the Deutsch algorithm



Basic ingredients

- Input in superposition (leading to some form of 'natural parallelism')
- An oracle for f taking the form of a controlled gate on the input
- A specific preparation of the first qubit (to induce some sort of interaction)

Input on demand

Usually, input is provided to the computational process once and for all in the form of a value, e.g., a binary string.

In this class of algorithms, however, the input is hidden, provided as a function

$$f : \mathbf{2}^n \longrightarrow \mathbf{2}^m$$

to which the process has limited access: it can only evaluate it on specific values.

A query selects a specific string $s \in \mathbf{2}^n$ and the oracle returns to the computational process the string $f(s) \in \mathbf{2}^m$.

The efficiency of an algorithm in this model is related to the number of queries it requires.

Input on demand

Examples of query problems

- Input: $f : \mathbf{2}^n \longrightarrow \mathbf{2}$
  Problem: Decide if there is a string s st $f(s) = 1$.
- Input: $f : \mathbf{2}^n \longrightarrow \mathbf{2}^m$
  Problem: Output the smallest string in the lexicographic order on $\mathbf{2}^m$ (assuming the function represents integers, the problem is to find their minimum)
- Input: $f : \mathbf{2}^n \longrightarrow \mathbf{2}$ with a promise (e.g. there is only one string s for which $f(s) = 1$.
  Problem: Find s

Oracles

Access to input is provided by an oracle.

Oracles, however, are not simply functions $f : \mathbf{2}^n \longrightarrow \mathbf{2}^m$ extended to linear maps $\mathbb{C}^{\mathbf{2}^n} \to \mathbb{C}^{\mathbf{2}^n}$, because often they do not correspond to unitary matrices.

They are build instead as

$$|x\rangle|y) \mapsto |x\rangle|y \oplus f(x)\rangle\rangle$$

as discussed in the previous lecture.

Example

Let $f : \{0,1\}^2 \longrightarrow \{0,1\}$ be such that $f(01) = 1$ and evaluates to 0 otherwise.
Oracle $U_f(|x\rangle|y\rangle) = |x\rangle|y \oplus f(x)\rangle$ can be tabulated as

$$\begin{array}{llll}
|00\rangle|0\rangle \mapsto |00\rangle|0\rangle & & |00\rangle|1\rangle \mapsto |00\rangle|1\rangle \\
|01\rangle|0\rangle \mapsto |01\rangle|1\rangle & & |01\rangle|1\rangle \mapsto |01\rangle|0\rangle \\
|10\rangle|0\rangle \mapsto |10\rangle|0\rangle & & |10\rangle|1\rangle \mapsto |10\rangle|1\rangle \\
|11\rangle|0\rangle \mapsto |11\rangle|0\rangle & & |11\rangle|1\rangle \mapsto |11\rangle|1\rangle
\end{array}$$

which corresponds to

$$\sum_{x\in\{0,1\}^2} |x\rangle\langle x|\, X^{f(x)} =$$

$$= |00\rangle\langle 00| \otimes I + |01\rangle\langle 01| \otimes X + |10\rangle\langle 10| \otimes I + |11\rangle\langle 11| \otimes I$$

Example

Or, in matrix format,

$$U_f = \begin{bmatrix} I & 0 & 0 & 0 \\ 0 & X & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Back to the Deutsch algorithm

What is new in quantum evaluation of Boolean functions is the ability to act on a superposition, e.g.

$$\sum_x |x\rangle|0\rangle \mapsto \sum_x |x\rangle|f(x)\rangle$$

i.e. all results are computed in a single execution

But also interesting is the effect of starting with $|-\rangle$:

$$\sum_x |x\rangle|-\rangle \mapsto \sum_x (-1)^{f(x)}|x\rangle|-\rangle$$

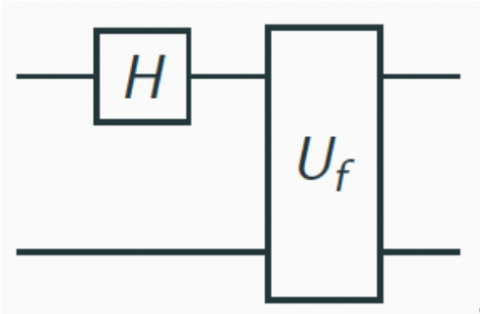
which indeed generates the suitable interference.

Revisiting the Deutsch algorithm

The oracle

$$|x\rangle|y\rangle \mapsto |x\rangle|y \oplus f(x)\rangle$$

leaves the first qubit unchanged, but modifies the second one.
Look again, however, how the oracle behaves in the circuit



behaves when processing $H|0\rangle|-\rangle = |+\rangle|-\rangle$, leading to

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

Pulling $(-1)^{f(0)}$ out of the sum, yields

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

Revisiting the Deutsch algorithm

Making the conditional structure explicit, this equivales to

$$\begin{cases} (\pm 1)|+\rangle|-\rangle & \text{if } f(0) \otimes f(1) = 0 \;\; (\text{i.e., } \; f(0) = f(1)) \\ (\pm 1)|-\rangle|-\rangle & \text{if } f(0) \otimes f(1) = 1 \;\; (\text{i.e., } \; f(0) \neq f(1)) \end{cases}$$

which later, after applying the final H to the first qubit, leads to

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

providing a solution to the Deutsch problem.

But now the second qubit remains unchanged and a phase factor is kicked back to the first one.

What's under the carpet?

Let's try to understand what explains this kick back behaviour. Clearly,

$$|b \oplus 1\rangle = |\neg b\rangle = X|b\rangle = X^1|b\rangle$$
$$|b \oplus 0\rangle = |b\rangle = I|b\rangle = X^0|b\rangle$$

that is,

$$|b \oplus c\rangle = X^c|b\rangle \text{ for } b, c \in 2$$

Therefore,

$$U_f(|a\rangle|b\rangle) = |a\rangle|b \oplus f(a)\rangle = |a\rangle X^{f(a)}|b\rangle \text{ for } a, b \in 2$$

and, by linearity (why?),

$$U_f(|a\rangle|v\rangle) = |a\rangle X^{f(a)}|v\rangle \text{ for for any vector } |v\rangle$$

What's under the carpet?

Making $|v\rangle = |-\rangle$, yields

$$U_f(|a\rangle|-\rangle) = |a\rangle X^{f(a)}|-\rangle$$

Finally

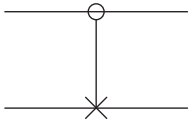
$$U_f(|a\rangle|-\rangle) = \begin{cases} |a\rangle I|-\rangle = |a\rangle|-\rangle & \text{if } f(a) = 0 \\ |a\rangle X|-\rangle = |a\rangle(-|-\rangle) = -|a\rangle|-\rangle & \text{if } f(a) = 1 \end{cases}$$

because

$$I|-\rangle = |-\rangle \text{ and } X|-\rangle = -|-\rangle$$

which means that $(-1, |-\rangle)$ is a eigen value/vector pair for operator X (and, similarly, $(1, |-\rangle)$ for I).

A even simpler example: the behaviour of cX



$$\overbrace{\begin{bmatrix} I & 0 \\ 0 & X \end{bmatrix}}^{cX} \quad \text{i.e.,} \quad |xy\rangle \mapsto |x, x \oplus y\rangle$$

where $|x\rangle$ is called the control qubit, and $|y\rangle$ the target qubit.
Clearly,

$$\begin{aligned} cX|0\rangle|\varphi\rangle &= |0\rangle I|\varphi\rangle \\ cX|1\rangle|\varphi\rangle &= |1\rangle X|\varphi\rangle \end{aligned}$$

Seen as an oracle, note that input is presented at the control qubit and output is produced on the target qubit.

A even simpler example: the behaviour of cX

Consider now a special case: prepare the target qubit with $|-\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}$ which, as seen before, is an eigenvector of both

- X (with $\lambda = -1$) and of I (with $\lambda = 1$)
- and, thus, $X \frac{|0\rangle - |1\rangle}{\sqrt{2}} = -1 \frac{|0\rangle - |1\rangle}{\sqrt{2}}$ and $I \frac{|0\rangle - |1\rangle}{\sqrt{2}} = 1 \frac{|0\rangle - |1\rangle}{\sqrt{2}}$

Therefore,

$$\begin{aligned} cX\,|1\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) &= |1\rangle \left(X \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)\right) \\ &= |1\rangle \left((-1) \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)\right) \\ &= -|1\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) \end{aligned}$$

while $cX\,|0\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) = |0\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)$

A even simpler example: the behaviour of cX

A phase (1 or -1, i.e., the corresponding eigenvalue)

jumps, or is kicked back

from the second (target) to the first (control) qubit where the input is presented.

This effect is suitably recorded in the following formulation of cX:

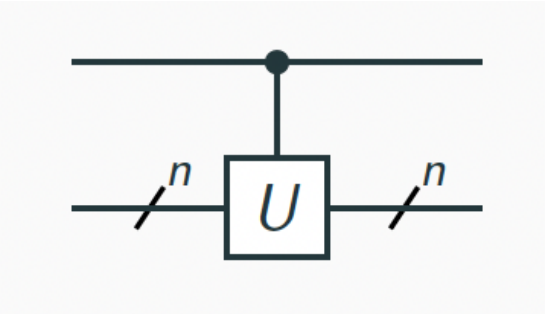
$$cX|b\rangle|-\rangle = (-1)^b|b\rangle|-\rangle \quad \text{with } b \in \mathbf{2}$$

Observe now that, through the kick-back effect, ouput arises in the control qubit, whereas the target qubit remains unchanged.
Example:

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

The phase kick-back pattern

This can be generalised to every controlled quantum operation:



Let v be an eigenvector of U with eigenvalue $e^{i\theta}$, (*i.e.* $Uv = e^{i\theta}v$). Thus,

$$
\begin{aligned}
& cU\Big((\alpha|0\rangle + \beta|1\rangle) \otimes v\Big) \\
= \; & cU(\alpha|0\rangle \otimes v + \beta|1\rangle \otimes v) \\
= \; & \alpha|0\rangle \otimes v + \beta|1\rangle \otimes Uv \\
= \; & \alpha|0\rangle \otimes v + \beta|1\rangle \otimes e^{i\theta}v \\
= \; & \alpha|0\rangle \otimes v + e^{i\theta}\beta|1\rangle \otimes v \\
= \; & (\alpha|0\rangle + e^{i\theta}\beta|1\rangle) \otimes v
\end{aligned}
$$

The phase kick-back pattern

- Global phase $e^{i\theta}$ (introduced to v) was 'kicked-back' as a relative phase in the control qubit
- Some information of U is now encoded in the control qubit

In general kicking-back such phases causes interference patterns that give away information about U.

Our two examples

Phase kick-back appears as

in the cX gate: $cX|b\rangle|-\rangle = (-1)^b|b\rangle|-\rangle$

in Deutsch algorithm: $U_f|x\rangle|-\rangle = (-1)^{f(x)}|x\rangle|-\rangle$

Our two examples

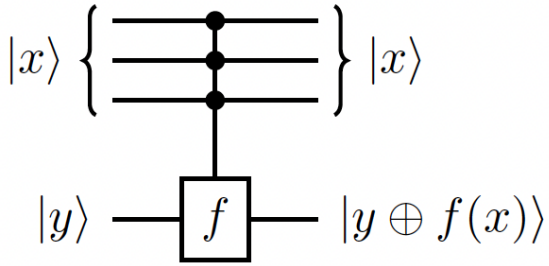
in the cX gate: $cX|b\rangle|-\rangle = (-1)^b|b\rangle|-\rangle$

in Deutsch algorithm: $U_f|x\rangle|-\rangle = (-1)^{f(x)}|x\rangle|-\rangle$

Note

The oracle in the Deutsch algorithm takes the form of a generalised cX:

$$\boxed{\sum_{x\in\{0,1\}^n} |x\rangle\langle x|\, X^{f(x)}}$$



The Bernstein-Vazirani algorithm

Let $2^n = \{0,1\}^n = \{0,1,2,\cdots 2^n - 1\}$ be the set of non-negative integers (represented as bit strings up to n bits).

The problem

Let s be an unknown non-negative integer less than 2^n, encoded as a bit string, and consider a function $f : \{0,1\}^n \to \{0,1\}$ which hides secret s as follows: $f(x) = x \cdot s$, where $\cdot$ is the bitwise product of x and s modulo 2. i.e.

$$x \cdot s \;=\; x_1 s_1 \oplus x_2 s_2 \oplus \cdots \oplus x_n s_n$$

Find s.

Note that juxtaposition abbreviates conjunction, i.e., $x_1 s_1 \;=\; x_1 \wedge s_1$

Setting the stage

Lemma

(1) *For $a, b \in 2$ the equation $(-1)^a(-1)^b = (-1)^{a \oplus b}$ holds.*

Proof sketch

Build a truth table for each case and compare the corresponding contents.

Lemma

(2) *For any three binary strings $x, a, b \in 2^n$ the equation $(x \cdot a) \oplus (x \cdot b) = x \cdot (a \oplus b)$ holds.*

Proof sketch

Follows from the fact that for any three bits $a, b, c \in 2$ the equation $(a \wedge b) \oplus (a \wedge c) = a \wedge (b \oplus c)$ holds.

Setting the stage

Lemma

(3) *For any element $|b\rangle$ in the computational basis of $\mathbb{C}^2$,*

$$H|b\rangle = \frac{1}{\sqrt{2}} \sum_{z \in 2} (-1)^{b \wedge z} |z\rangle$$

Proof sketch

Build a truth table and compare the corresponding contents.

Theorem

(1) *For any element $|b\rangle$ in the computational basis of $\mathbb{C}^{2^n}$,*

$$H^{\otimes n}|b\rangle = \frac{1}{\sqrt{2^n}} \sum_{z \in 2^n} (-1)^{b \cdot z} |z\rangle$$

Proof sketch

Follows by induction on the size of n.

The Bernstein-Vazirani algorithm

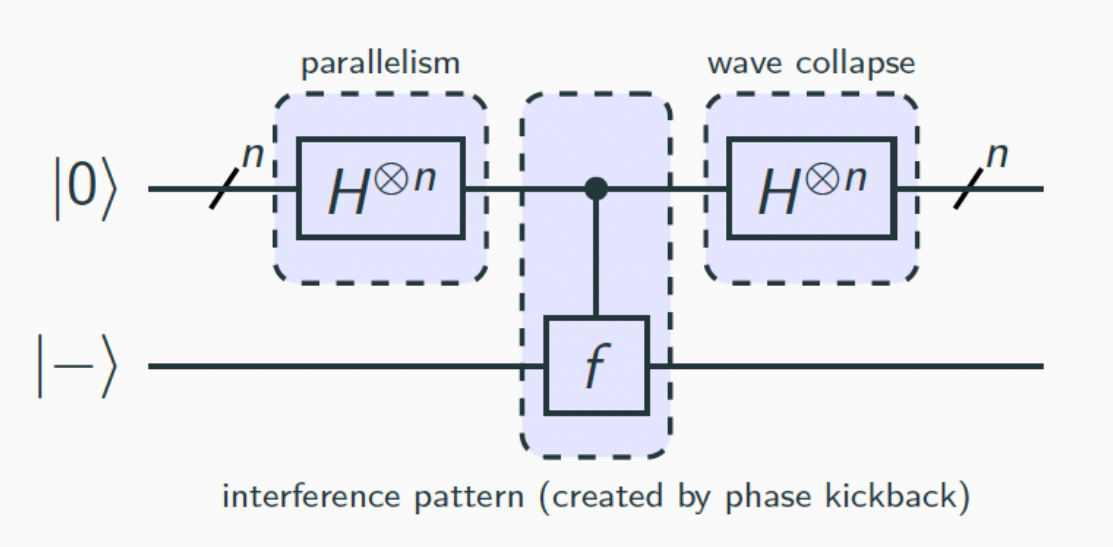
How many times f has to be called to determine s?

- Classically, we run f n-times by computing

$$f(1\ldots0) = (s_1 \wedge 1) \oplus \cdots \oplus (s_n \wedge 0) = s_1$$
$$\vdots$$
$$f(0\ldots1) = (s_1 \wedge 0) \oplus \cdots \oplus (s_n \wedge 1) = s_n$$

- With a quantum algorithm, we may discover s by running f only once

The circuit


parallelism
wave collapse
$|0\rangle$
$|-\rangle$
n
$H^{\otimes n}$
f
$H^{\otimes n}$
n
interference pattern (created by phase kickback)

The computation

$$
\begin{aligned}
& H^{\otimes n}|0\rangle|-\rangle \\
&= \tfrac{1}{\sqrt{2^n}} \textstyle\sum_{z\in 2^n} |z\rangle|-\rangle && \{\text{Theorem 1}\} \\
&\overset{U_f}{\mapsto} \quad \tfrac{1}{\sqrt{2^n}} \textstyle\sum_{z\in 2^n} (-1)^{f(z)}|z\rangle|-\rangle && \{\text{Defn. } U_f\} \\
&\overset{H^{\otimes n}\otimes I}{\mapsto} \quad \tfrac{1}{2^n} \textstyle\sum_{z\in 2^n} (-1)^{f(z)} \Big(\sum_{z'\in 2^n} (-1)^{z\cdot z'}|z'\rangle \Big)|-\rangle && \{\text{Theorem 1}\} \\
&= \tfrac{1}{2^n} \textstyle\sum_{z\in 2^n} \sum_{z'\in 2^n} (-1)^{(z\cdot s)\oplus(z\cdot z')}|z'\rangle|-\rangle && \{\text{Lemma 1 and defn. } f\} \\
&= \tfrac{1}{2^n} \textstyle\sum_{z\in 2^n} \sum_{z'\in 2^n} (-1)^{z\cdot(s\oplus z')}|z'\rangle|-\rangle && \{\text{Lemma 2}\} \\
&= |s\rangle|-\rangle && \{\text{Why?}\}
\end{aligned}
$$

Why?

$$\cdots = \frac{1}{2^n} \sum_{z \in 2^n} \sum_{z' \in 2^n} (-1)^{z \cdot (s \oplus z')} |z'\rangle |-\rangle = \cdots$$

For each binary string $s \oplus z'$, $\frac{1}{2^n} \sum_{z=0}^{2^n-1} (-1)^{z \cdot (s \oplus z')}$ is 1 iff $(s \oplus z') = 0$, which happens only if $s = z'$.

In all other cases $\frac{1}{2^n} \sum_{z=0}^{2^n-1} (-1)^{z \cdot (s \oplus z')}$ is 0.

Thus, the only non-zero amplitude is the one associated with s.

Why?

The conclusion comes from the following result: for each binary string x,

$$\frac{1}{2^n}\sum_{z=0}^{2^n-1}(-1)^{z\cdot x} = \begin{cases} 1 & \text{if } x = 0^n \\ 0 & \text{if } x \neq 0^n \end{cases}$$

Proof

- for $x = 0$, $\frac{1}{2^n}\sum_{z=0}^{2^n-1}(-1)^{z\cdot x} = \frac{1}{2^n}\sum_{z=0}^{2^n-1} 1 = 1$.
- for $x \neq 0$, half of the 2^n factors in the sum will be -1 and the other half 1, thus summing up to 0. This is because the value of $z \cdot x$ is 1 for an odd number of coincidences of 1 bits in both arguments, and 0 otherwise, and z spans all numbers from 0 to $2^n - 1$

Why?

Alternatively, consider the probability of measuring s at the end of the computation:

$$\left|\frac{1}{2^n}\sum_{z\in 2^n}(-1)^{z\cdot(s\oplus s)}\right|^2$$
$$=\left|\frac{1}{2^n}\sum_{z\in 2^n}(-1)^{z\cdot 0}\right|^2$$
$$=\left|\frac{1}{2^n}\sum_{z\in 2^n}1\right|^2$$
$$=\left|\frac{2^n}{2^n}\right|^2$$
$$=1$$

This means that somehow all values yielding wrong answers were completely cancelled.

Deutsch-Josza

The Problem

Take a function $f : \{0,1\}^n \to \{0,1\}$, which is known to be either constant or balanced.

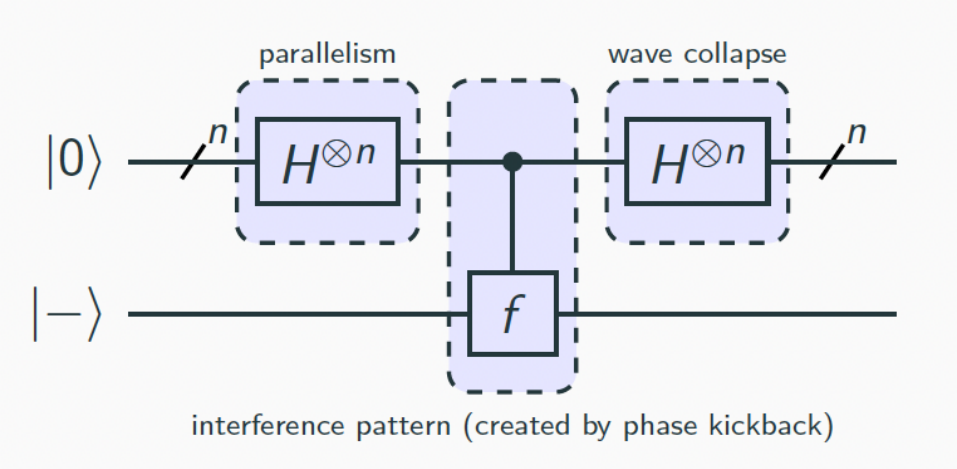
Find out which case holds.

Classically, we evaluate half of the inputs ($\frac{2^n}{2} = 2^{n-1}$), evaluate one more and run the decision procedure,

- output always the same $\Longrightarrow$ constant
- otherwise $\Longrightarrow$ balanced

which requires running f $2^{n-1} + 1$ times.
A quantum algorithm replies by running f only once.

The circuit



Actually, the algorithm is the same used to solve the Bernstein-Varziani problem.

More precisely, the problem is solved with the Bernstein-Varziani algorithm for a function satisfying the Deutsch-Joza promisse.

The computation

$$
\begin{aligned}
& H^{\otimes n}|0\rangle|-\rangle \\
& = \tfrac{1}{\sqrt{2^n}} \textstyle\sum_{z\in 2^n} |z\rangle|-\rangle && \{\text{Theorem 1}\} \\
& \overset{U_f}{\mapsto} \quad \tfrac{1}{\sqrt{2^n}} \textstyle\sum_{z\in 2^n} (-1)^{f(z)}|z\rangle|-\rangle && \{\text{Defn. } U_f\} \\
& \overset{H^{\otimes n}\otimes I}{\mapsto} \quad \underbrace{\tfrac{1}{2^n} \textstyle\sum_{z\in 2^n} (-1)^{f(z)} \left(\textstyle\sum_{z'\in 2^n} (-1)^{z\cdot z'}|z'\rangle \right)}_{\square \text{ upper qubits}} |-\rangle && \{\text{Theorem 1}\}
\end{aligned}
$$

Developing $\square$ by case distinction

f is constant

$$\frac{1}{2^n}\sum_{z\in 2^n}(-1)^{f(z)}\left(\sum_{z'\in 2^n}(-1)^{z\cdot z'}|z'\rangle\right)$$
$$= \frac{1}{2^n}(\pm 1)\sum_{z\in 2^n}\left(\sum_{z'\in 2^n}(-1)^{z\cdot z'}|z'\rangle\right)$$

Therefore, at state $|0\rangle$ is

$\boxed{f \text{ is constant at } 1}$ $\rightsquigarrow$ $\frac{-(2^n)}{2^n}|\mathbf{0}\rangle = -|\mathbf{0}\rangle$

$\boxed{f \text{ is constant at } 0}$ $\rightsquigarrow$ $\frac{(2^n)}{2^n}|\mathbf{0}\rangle = |\mathbf{0}\rangle$

Developing $\square$ by case distinction

Actually the probability of measuring $|0\rangle$ at the end given by

$$\left|\frac{1}{2^n}(\pm 1)\sum_{z\in 2^n}(-1)^{z\cdot 0}\right|^2$$
$$= \left|\frac{1}{2^n}(\pm 1)\sum_{z\in 2^n} 1\right|^2$$
$$= \left|\frac{2^n}{2^n}\right|^2$$
$$= 1$$

So if f is constant we measure $|0\rangle$ with probability 1.

Developing $\square$ by case distinction

f is balanced

$$\frac{1}{2^n}\sum_{z\in 2^n}(-1)^{f(z)}\left(\sum_{z'\in 2^n}(-1)^{z\cdot z'}|z'\rangle\right)$$

$$= \frac{1}{2^n}\left(\sum_{z\in 2^n, f(z)=0}(-1)^{f(z)}\left(\sum_{z'\in 2^n}(-1)^{z\cdot z'}|z'\rangle\right)\right.$$

$$\left.+\sum_{z\in 2^n, f(z)=1}(-1)^{f(z)}\left(\sum_{z'\in 2^n}(-1)^{z\cdot z'}|z'\rangle\right)\right)$$

$$= \frac{1}{2^n}\left(\sum_{z\in 2^n, f(z)=0}\left(\sum_{z'\in 2^n}(-1)^{z\cdot z'}|z'\rangle\right)\right.$$

$$\left.+\sum_{z\in 2^n, f(z)=1}(-1)\left(\sum_{z'\in 2^n}(-1)^{z\cdot z'}|z'\rangle\right)\right)$$

Developing □ by case distinction

Probability of measuring $|0\rangle$ at the end given by

$$\left|\frac{1}{2^n}\left(\sum_{z\in 2^n, f(z)=0}(-1)^{z\cdot 0}+\sum_{z\in 2^n, f(z)=1}(-1)(-1)^{z\cdot 0}\right)\right|^2$$
$$=\left|\frac{1}{2^n}\left(\sum_{z\in 2^n, f(z)=0}1+\sum_{z\in 2^n, f(z)=1}(-1)\right)\right|^2$$
$$=\left|\frac{1}{2^n}\left(\sum_{z\in 2^n, f(z)=0}1-\sum_{z\in 2^n, f(z)=1}1\right)\right|^2$$
$$=0$$

So if f is balanced we measure $|0\rangle$ with probability 0

Concluding

Deutsch problem

Classically, need to run f twice. With a quantum algorithm once is enough.

Bernstein-Varziani problem

Classically, need to run f n times. With a quantum algorithm once is enough.

Deutsch-Joza problem

Classically, need to evaluate half of the inputs ($\frac{2^n}{2} = 2^{n-1}$), evaluate one more and run the decision procedure,

- output always the same $\Longrightarrow$ constant
- otherwise $\Longrightarrow$ balanced

With a quantum algorithm once is enough.