Università degli Studi della Campania Luigi Vanvitelli

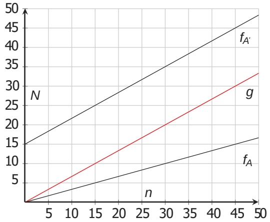
Future Computing Architecture and Programming Paradigms

Quantum Complexity

Estimating the run time of an algorithm

Big-O Notation

Let $f_A : \mathbb{N} \to \mathbb{N}$ be a function that returns the number of elementary calculation steps for an algorithm A, given an input with size n. We write $f_A \in O(g(n))$ if f grows asymptotically as fast or slower than g.



Common asymptotic functions

Function	designation	example algorithm
1	constant	calculate mod 2

Common asymptotic functions

Function	designation	example algorithm
1	constant	calculate mod 2
$log(n)$	logarithmic	binary search (in sorted database)

Common asymptotic functions

Function	designation	example algorithm
1	constant	calculate mod 2
$log(n)$	logarithmic	binary search (in sorted database)
n	linear	search in unsorted data

Common asymptotic functions

Function	designation	example algorithm
1	constant	calculate mod 2
$log(n)$	logarithmic	binary search (in sorted database)
n	linear	search in unsorted data
$n\,log(n)$	superlinear	merge sort

Common asymptotic functions

Function	designation	example algorithm
1	constant	calculate mod 2
$log(n)$	logarithmic	binary search (in sorted database)
n	linear	search in unsorted data
$n\,log(n)$	superlinear	merge sort
n^2	quadratic	multiplication of integers

Common asymptotic functions

Function	designation	example algorithm
1	constant	calculate mod 2
$log(n)$	logarithmic	binary search (in sorted database)
n	linear	search in unsorted data
$n\,log(n)$	superlinear	merge sort
n^2	quadratic	multiplication of integers
n^3	cubic	matrix multiplication

Big-O notation

Common asymptotic functions

Function	designation	example algorithm
1	constant	calculate mod 2
$log(n)$	logarithmic	binary search (in sorted database)
n	linear	search in unsorted data
$n\,log(n)$	superlinear	merge sort
n^2	quadratic	multiplication of integers
n^3	cubic	matrix multiplication
n^k	polynomial ($k \in \mathbb{N}$ fixed)	

Common asymptotic functions

Function	designation	example algorithm
1	constant	calculate mod 2
$log(n)$	logarithmic	binary search (in sorted database)
n	linear	search in unsorted data
$n\,log(n)$	superlinear	merge sort
n^2	quadratic	multiplication of integers
n^3	cubic	matrix multiplication
n^k	polynomial ($k \in \mathbb{N}$ fixed)	
2^n	exponential	naive calculation of the n-th Fibonacci number

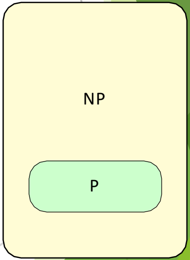
Complexity classes for decision problems

- **P** contains all problems **solvable** in polynomial time (on a deterministic Turing machine).

P

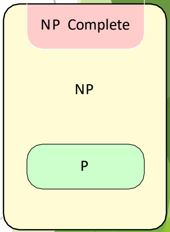
Complexity classes for decision problems

- **P** contains all problems **solvable** in polynomial time (on a deterministic Turing machine).
- **NP** contains all problems **verifiable** in polynomial time (or solvable on a non-deterministic Turing machine)



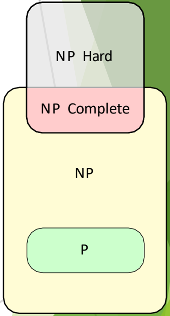
Complexity classes for decision problems

- **P** contains all problems **solvable** in polynomial time (on a deterministic Turing machine).
- **NP** contains all problems **verifiable** in polynomial time (or solvable on a non-deterministic Turing machine)
- **NP-complete**: The problem is in **NP** and **every** problem in NP can be reduced to it in polynomial time.



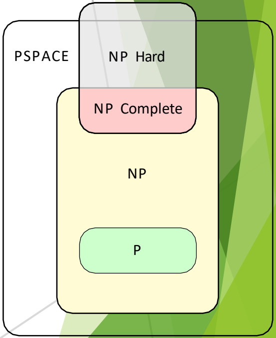
Complexity classes for decision problems

- **P** contains all problems **solvable** in polynomial time (on a deterministic Turing machine).
- **NP** contains all problems **verifiable** in polynomial time (or solvable on a non-deterministic Turing machine)
- **NP-complete**: The problem is in **NP** and **every** problem in NP can be reduced to it in polynomial time.
- **NP-hard**: **Every** problem in NP can be reduced to the problem in polynomial time.



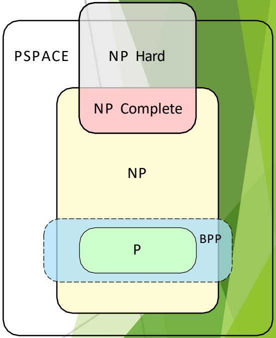
Complexity classes for decision problems

- **P** contains all problems **solvable** in polynomial time (on a deterministic Turing machine).
- **NP** contains all problems **verifiable** in polynomial time (or solvable on a non-deterministic Turing machine)
- **NP-complete**: The problem is in **NP** and **every** problem in NP can be reduced to it in polynomial time.
- **NP-hard**: **Every** problem in NP can be reduced to the problem in polynomial time.
- **PSPACE**: The problem is solvable in polynomial **space**.



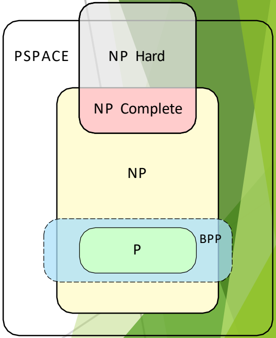
Bounded-error probabilistic polynomial time (BPP)

- Algorithms are allowed to return a wrong result with probability $< \frac{1}{2}$.



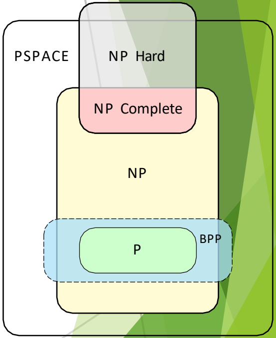
Bounded-error probabilistic polynomial time (BPP)

- Algorithms are allowed to return a wrong result with probability $< \frac{1}{2}$.
- Errors have to be made by chance → *randomized* algorithm.



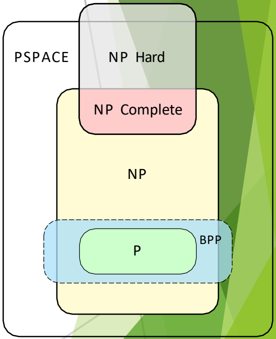
Bounded-error probabilistic polynomial time (BPP)

- Algorithms are allowed to return a wrong result with probability $< \frac{1}{2}$.
- Errors have to be made by chance → *randomized* algorithm.
- Error probability can be arbitrarily reduced via majority vote of multiple iterations.



Bounded-error probabilistic polynomial time (BPP)

- Algorithms are allowed to return a wrong result with probability $< \frac{1}{2}$.
- Errors have to be made by chance → *randomized* algorithm.
- Error probability can be arbitrarily reduced via majority vote of multiple iterations.
- BPP = *frontier of feasibility* for classical computers.



Bounded-error quantum polynomial time (BQP)

BQP

A decision problem E is in the class BQP if there is an algorithm A:

- If the correct solution $E(x) = 1$ for an input x, the algorithm generates the result $A(x) = 1$ with probability greater than $\frac{1}{2}$.
- If the correct solution $E(x) = 0$ for an input x, the algorithm generates the result $A(x) = 0$ with probability greater than $\frac{1}{2}$.
- The solution can be calculated with uniform quantum circuits of polynomial size.

Back to BQP

$\Rightarrow$ NOT, AND, OR are a classical universal gate set.

Back to BQP

$\Rightarrow$ NOT, AND, OR are a classical universal gate set.

$\Rightarrow$ Toffoli gate is reversible and can be implemented on quantum computer.

Back to BQP

⇒ NOT, AND, OR are a classical universal gate set.

⇒ Toffoli gate is reversible and can be implemented on quantum computer.

⇒ Every classical circuit can be converted to a quantum circuit with only **linear overhead**.

Back to BQP

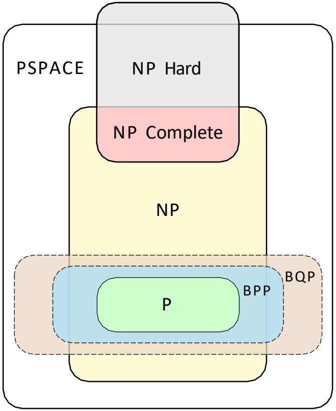
⇒ NOT, AND, OR are a classical universal gate set.

⇒ Toffoli gate is reversible and can be implemented on quantum computer.

⇒ Every classical circuit can be converted to a quantum circuit with only **linear overhead**.

⇒ $P \subseteq BQP$

Back to BQP



Oracle function

- Complexity-theoretic oracle solves problem in a **single step** $O(1)$.

Oracle function

- Complexity-theoretic oracle solves problem in a **single step** $O(1)$.
- Functionality is hidden from outside (Black box model)

Oracle function

- Complexity-theoretic oracle solves problem in a **single step** $O(1)$.
- Functionality is hidden from outside (Black box model)
- Quantum oracle $U_f : |\psi\rangle - \boxed{U_f} - |\psi'\rangle$

Oracle function

- Complexity-theoretic oracle solves problem in a **single step** $O(1)$.
- Functionality is hidden from outside (Black box model)
- Quantum oracle U_f : $|\psi\rangle - \boxed{U_f} - |\psi'\rangle$
- Quantum computing allows to reduce calls to an oracle!

Interference changes the behavior of oracles

Hadamard "Sandwich": $$|\psi\rangle \;-\!\!-\boxed{H}\!-\!\!-\boxed{U_f}\!-\!\!-\boxed{H}\!-\!\!-\; |\psi'\rangle$$

Interference changes the behavior of oracles

Hadamard “Sandwich”: $|\psi\rangle - \boxed{H} - \boxed{X} - \boxed{H} - |\psi'\rangle$

Interference changes the behavior of oracles

Hadamard "Sandwich": $|\psi\rangle - \boxed{H} - \boxed{X} - \boxed{H} - |\psi'\rangle$

$|\psi\rangle = |0\rangle$:

- $|\psi_1\rangle = H\,|0\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right)$

Interference changes the behavior of oracles

Hadamard "Sandwich": $|\psi\rangle - \boxed{H} - \boxed{X} - \boxed{H} - |\psi'\rangle$

$|\psi\rangle = |0\rangle$:

- $|\psi_1\rangle = H\,|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$
- $|\psi_2\rangle = X\,|\psi_1\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$

Interference changes the behavior of oracles

Hadamard "Sandwich": $|\psi\rangle - \boxed{H} - \boxed{X} - \boxed{H} - |\psi'\rangle$

$|\psi\rangle = |0\rangle$:

- $|\psi_1\rangle = H\,|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$
- $|\psi_2\rangle = X\,|\psi_1\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$
- $|\psi'\rangle = H\,|\psi_2\rangle$

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

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

Interference changes the behavior of oracles

Hadamard "Sandwich": $|\psi\rangle - \boxed{H} - \boxed{X} - \boxed{H} - |\psi'\rangle$

$|\psi\rangle = |0\rangle$:

- $|\psi_1\rangle = H\,|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$
- $|\psi_2\rangle = X\,|\psi_1\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$
- $|\psi'\rangle = H\,|\psi_2\rangle$

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

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

$|\psi\rangle = |1\rangle$:

- $|\psi_1\rangle = H\,|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$

Interference changes the behavior of oracles

Hadamard "Sandwich": $|\psi\rangle - \boxed{H} - \boxed{X} - \boxed{H} - |\psi'\rangle$

$|\psi\rangle = |0\rangle$:

- $|\psi_1\rangle = H\,|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$
- $|\psi_2\rangle = X\,|\psi_1\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$
- $|\psi'\rangle = H\,|\psi_2\rangle$

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

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

$|\psi\rangle = |1\rangle$:

- $|\psi_1\rangle = H\,|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$
- $|\psi_2\rangle = X\,|\psi_1\rangle = -\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$

Interference changes the behavior of oracles

Hadamard "Sandwich": $|\psi\rangle - \boxed{H} - \boxed{X} - \boxed{H} - |\psi'\rangle$

$|\psi\rangle = |0\rangle$:

- $|\psi_1\rangle = H\,|0\rangle = \frac{1}{\sqrt{2}}\,(|0\rangle + |1\rangle)$
- $|\psi_2\rangle = X\,|\psi_1\rangle = \frac{1}{\sqrt{2}}\,(|0\rangle + |1\rangle)$
- $|\psi'\rangle = H\,|\psi_2\rangle$

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

$= \frac{1}{2}\,(|0\rangle + |1\rangle + |0\rangle - |1\rangle) = |0\rangle$

$|\psi\rangle = |1\rangle$:

- $|\psi_1\rangle = H\,|1\rangle = \frac{1}{\sqrt{2}}\,(|0\rangle - |1\rangle)$
- $|\psi_2\rangle = X\,|\psi_1\rangle = -\frac{1}{\sqrt{2}}\,(|0\rangle - |1\rangle)$
- $|\psi'\rangle = H\,|\psi_2\rangle$

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

$= -\frac{1}{2}\,(|0\rangle + |1\rangle - |0\rangle + |1\rangle) = -\,|1\rangle$

Quantum oracles

Behavior of gates in Hadamard sandwich

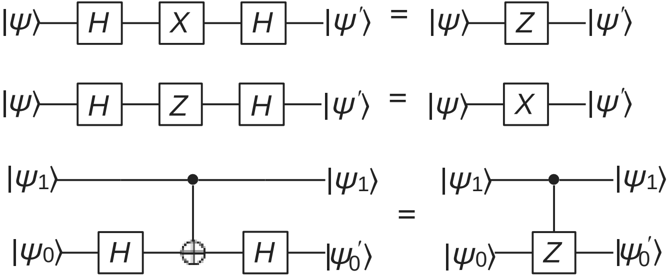
$$|\psi\rangle - \boxed{H} - \boxed{X} - \boxed{H} - |\psi'\rangle \; = \; |\psi\rangle - \boxed{Z} - |\psi'\rangle$$

Behavior of gates in Hadamard sandwich

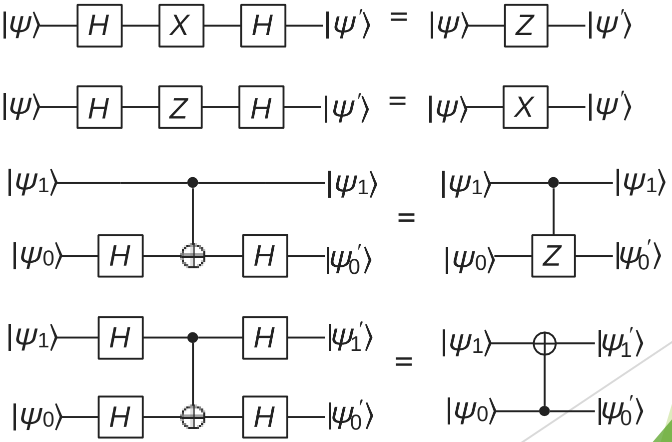
$$|\psi\rangle - \boxed{H} - \boxed{X} - \boxed{H} - |\psi'\rangle \;=\; |\psi\rangle - \boxed{Z} - |\psi'\rangle$$

$$|\psi\rangle - \boxed{H} - \boxed{Z} - \boxed{H} - |\psi'\rangle \;=\; |\psi\rangle - \boxed{X} - |\psi'\rangle$$

Behavior of gates in Hadamard sandwich



Behavior of gates in Hadamard sandwich



Summary

$\Rightarrow$ With interference we can find out information of an oracle which we would not get in the classical case.

Algorithm of Deutsch

- Balanced vs. constant functions

▶Given an oracle $f : \mathrm{B} \rightarrow \mathrm{B}$ which maps a binary input to a binary output. Find out whether the function is

Algorithm of Deutsch

- Balanced vs. constant functions

- Given an oracle $f : \mathrm{B} \to \mathrm{B}$ which maps a binary input to a binary output. Find out whether the function is
  - balanced, i.e. f returns 0 for one input and 1 for the other one.

Algorithm of Deutsch

- Balanced vs. constant functions

- Given an oracle $f : \mathrm{B} \rightarrow \mathrm{B}$ which maps a binary input to a binary output. Find out whether the function is
  - balanced, i.e. f returns 0 for one input and 1 for the other one.
  - constant, i.e. f returns the same number for both inputs.

Algorithm of Deutsch

- Balanced vs. constant functions

  - Given an oracle $f : \mathrm{B} \to \mathrm{B}$ which maps a binary input to a binary output. Find out whether the function is

    - balanced, i.e. f returns 0 for one input and 1 for the other one.
    - constant, i.e. f returns the same number for both inputs.

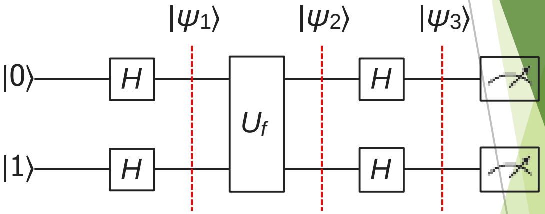
- $\Rightarrow$ Classically we need two calls to the oracle for solving the problem.

Algorithm of Deutsch

- Balanced vs. constant functions
  - Given an oracle $f : \mathrm{B} \to \mathrm{B}$ which maps a binary input to a binary output. Find out whether the function is
    - balanced, i.e. f returns 0 for one input and 1 for the other one.
    - constant, i.e. f returns the same number for both inputs.
- $\Rightarrow$ Classically we need two calls to the oracle for solving the problem.
- $\Rightarrow$ On a quantum computer we need only one!

Algorithm of Deutsch

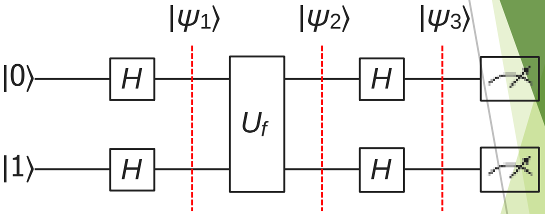
$$U_f : |x, y\rangle \mapsto |x, y \oplus f(x)\rangle$$



- $|\psi_1\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \cdot \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = \frac{1}{2}(|0\rangle|0\rangle - |0\rangle|1\rangle + |1\rangle|0\rangle - |1\rangle|1\rangle)$

Algorithm of Deutsch

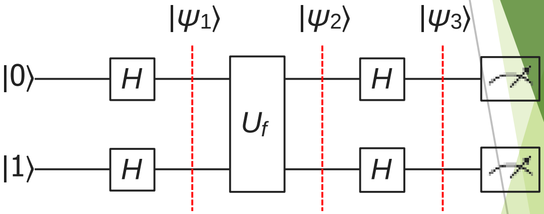
$$U_f : |x, y\rangle \mapsto |x, y \oplus f(x)\rangle$$



- $|\psi_1\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \cdot \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = \frac{1}{2}(|0\rangle|0\rangle - |0\rangle|1\rangle + |1\rangle|0\rangle - |1\rangle|1\rangle)$
- $|\psi_2\rangle = \frac{1}{2}(|0\rangle|0 \oplus f(0)\rangle - |0\rangle|1 \oplus f(0)\rangle + |1\rangle|0 \oplus f(1)\rangle - |1\rangle|1 \oplus f(1)\rangle)$

Algorithm of Deutsch

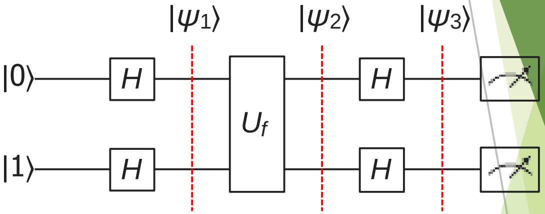
$$U_f : |x, y\rangle \mapsto |x, y \oplus f(x)\rangle$$



- $|\psi_1\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \cdot \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = \frac{1}{2}(|0\rangle|0\rangle - |0\rangle|1\rangle + |1\rangle|0\rangle - |1\rangle|1\rangle)$
- $|\psi_2\rangle = \frac{1}{2}(|0\rangle|f(0)\rangle - |0\rangle|1 \oplus f(0)\rangle + |1\rangle|f(1)\rangle - |1\rangle|1 \oplus f(1)\rangle)$

Algorithm of Deutsch

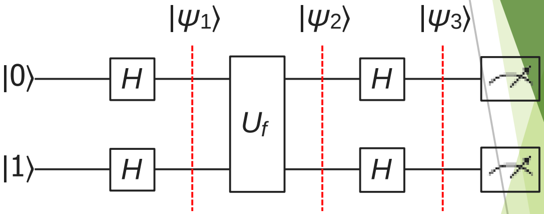
$$U_f : |x, y\rangle \mapsto |x, y \oplus f(x)\rangle$$



- $|\psi_1\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \cdot \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = \frac{1}{2}(|0\rangle|0\rangle - |0\rangle|1\rangle + |1\rangle|0\rangle - |1\rangle|1\rangle)$
- $|\psi_2\rangle = \frac{1}{2}(|0\rangle(|f(0)\rangle - |1 \oplus f(0)\rangle) + |1\rangle(|f(1)\rangle - |1 \oplus f(1)\rangle))$

Algorithm of Deutsch

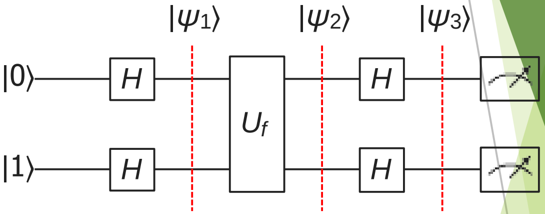
$$U_f : |x, y\rangle \mapsto |x, y \oplus f(x)\rangle$$



- $|\psi_1\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \cdot \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = \frac{1}{2}(|0\rangle|0\rangle - |0\rangle|1\rangle + |1\rangle|0\rangle - |1\rangle|1\rangle)$
- $|\psi_2\rangle = \frac{1}{2}\left[|0\rangle(-1)^{f(0)}(|0\rangle - |1\rangle) + |1\rangle(-1)^{f(1)}(|0\rangle - |1\rangle)\right]$

Algorithm of Deutsch

$$U_f : |x, y\rangle \mapsto |x, y \oplus f(x)\rangle$$



- $|\psi_1\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \cdot \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = \frac{1}{2}(|0\rangle|0\rangle - |0\rangle|1\rangle + |1\rangle|0\rangle - |1\rangle|1\rangle)$
- $|\psi_2\rangle = \frac{1}{2}\left[|0\rangle(-1)^{f(0)}(|0\rangle - |1\rangle) + |1\rangle(-1)^{f(1)}(|0\rangle - |1\rangle)\right]$

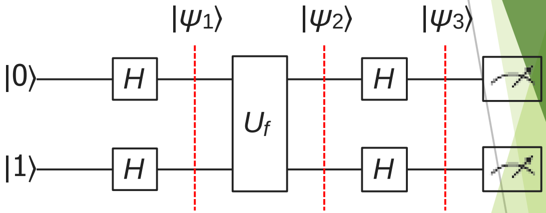
$$= \frac{1}{2}\left[(-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle\right] \cdot (|0\rangle - |1\rangle)$$

Algorithm of Deutsch

Case U_f constant

State of upper qubit:

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



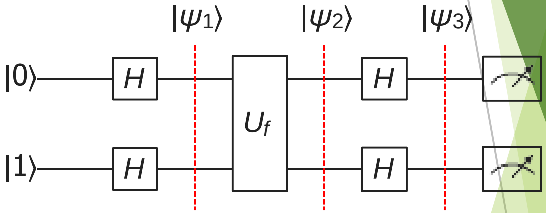
- It applies: $(-1)^{f(0)} = (-1)^{f(1)}$, i.e. the state is $\pm\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$

Algorithm of Deutsch

Case U_f constant

State of upper qubit:

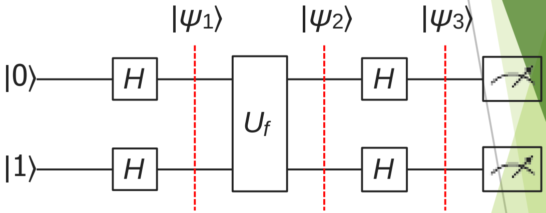
$$\frac{1}{\sqrt{2}}\left((-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle\right)$$



- It applies: $(-1)^{f(0)} = (-1)^{f(1)}$, i.e. the state is $\pm \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$

$\Rightarrow$ Hadamard Gate transforms state to $\pm\ |0\rangle$

Case U_f constant



State of upper qubit:

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

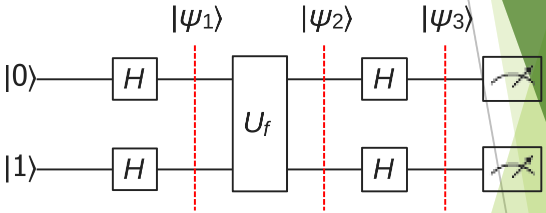
- It applies: $(-1)^{f(0)} = (-1)^{f(1)}$, i.e. the state is $\pm \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$

$\Rightarrow$ Hadamard Gate transforms state to $\pm\ |0\rangle$

$\Rightarrow$ We always measure $|0\rangle$ on the upper qubit.

Algorithm of Deutsch

Case U_f balanced



State of upper qubit:

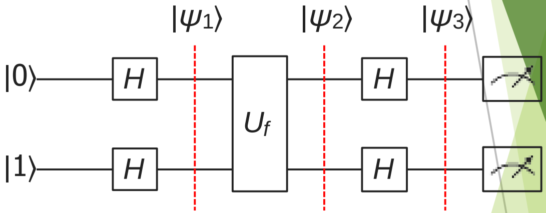
$$\frac{1}{\sqrt{2}}\left((-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle\right)$$

- It applies: $(-1)^{f(0)} \neq (-1)^{f(1)}$, i.e. the state is $\pm\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$

Case U_f balanced

State of upper qubit:

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



- It applies: $(-1)^{f(0)} \neq (-1)^{f(1)}$, i.e. the state is $\pm \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$

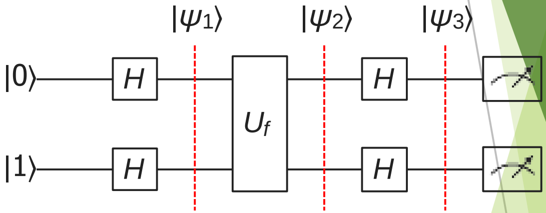
$\Rightarrow$ Hadamard Gate transforms state to $\pm\, |1\rangle$

Algorithm of Deutsch

Case U_f **balanced**

State of upper qubit:

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



- It applies: $(-1)^{f(0)} \neq (-1)^{f(1)}$, i.e. the state is $\pm\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$

⇒ Hadamard Gate transforms state to $\pm\,|1\rangle$

⇒ We always measure $|1\rangle$ on the upper qubit.

Summary

- Quantum oracle U_f is only applied **once** to determine whether f is constant or balanced.

Summary

- Quantum oracle U_f is only applied **once** to determine whether f is constant or balanced.
- Can be **extended** to boolean functions with **arbitrary input size**.

Summary

- Quantum oracle U_f is only applied **once** to determine whether f is constant or balanced.
- Can be **extended** to boolean functions with **arbitrary input size**.
- **Theoretical** speedup over classical counterpart, but small **practical** value.

Summary

- Quantum oracle U_f is only applied **once** to determine whether f is constant or balanced.
- Can be **extended** to boolean functions with **arbitrary input size**.
- **Theoretical** speedup over classical counterpart, but small **practical** value.
- However, most quantum algorithms with provable speedup rely on **quantum oracles** in combination with **interference**.