Quantum Computation and Information Theory

Vwani Roychowdhury

Electrical Engineering Department

UCLA

Classical Models of Computation

— Turing Machines

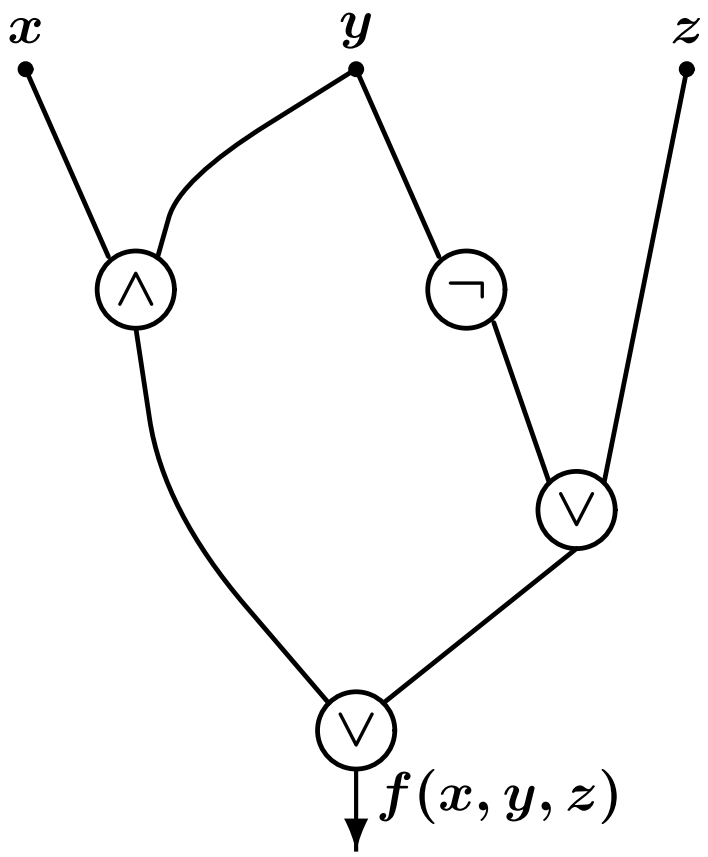
— Boolean Circuits

gates: AND $\bigwedge$, OR $\bigvee$, NOT $\neg$

size of the circuit: the number of gates

A circuit with n inputs computes a Boolean function

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



x	y	z	$f(x,y,z)$
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

The basis $\{$NOT, AND, OR$\}$ is complete (or universal)

i.e., for any Boolean function $f : \{0,1\}^n \longrightarrow \{0,1\}$ there is a Boolean circuit with

NOT, AND, OR

gates computing this function

Shannon Bound: For almost all such functions f, the optimal Boolean circuit computing f has

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

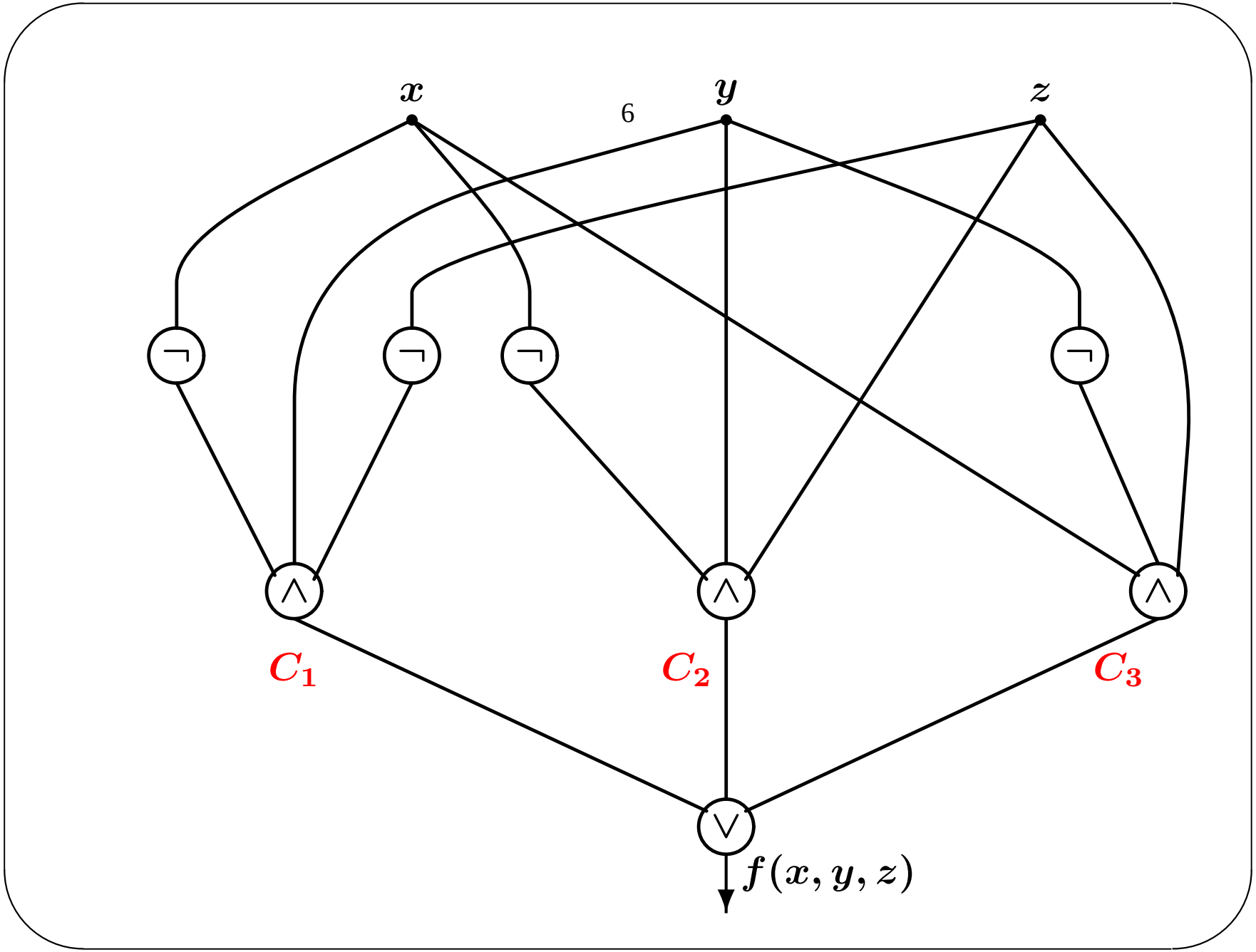
gates

universality of Boolean basis, an example:

$$f : \{0,1\}^3 \longrightarrow \{0,1\}$$

x	y	z	$f(x,y,z)$	
0	0	0	0	
0	0	1	0	
0	1	0	1	C_1
0	1	1	1	C_2
1	0	0	0	
1	0	1	1	C_3
1	1	0	0	
1	1	1	0	

$$f(x,y,z) = \underbrace{(\neg x \wedge y \wedge \neg z)}_{C_1} \vee \underbrace{(\neg x \wedge y \wedge z)}_{C_2} \vee \underbrace{(x \wedge \neg y \wedge z)}_{C_3}$$


x
y
z
¬
¬
¬
¬
∧
∧
∧
C1
C2
C3
∨
f(x, y, z)

Solving problems with variable sizes by circuits

$N \in \{0, 1, 2, 3, \dots\}$, $\qquad f(N) = \begin{cases} 1 & \text{if } N \text{ is prime,} \\ 0 & \text{otherwise.} \end{cases}$

size(N) = number of bits in binary expansion of N

size(5) = size$(101) = 3$

size(13) = size$(1101) = 4$

computing f by circuits means that there is a sequence

$$C_1, C_2, C_3, \dots, C_n, \dots$$

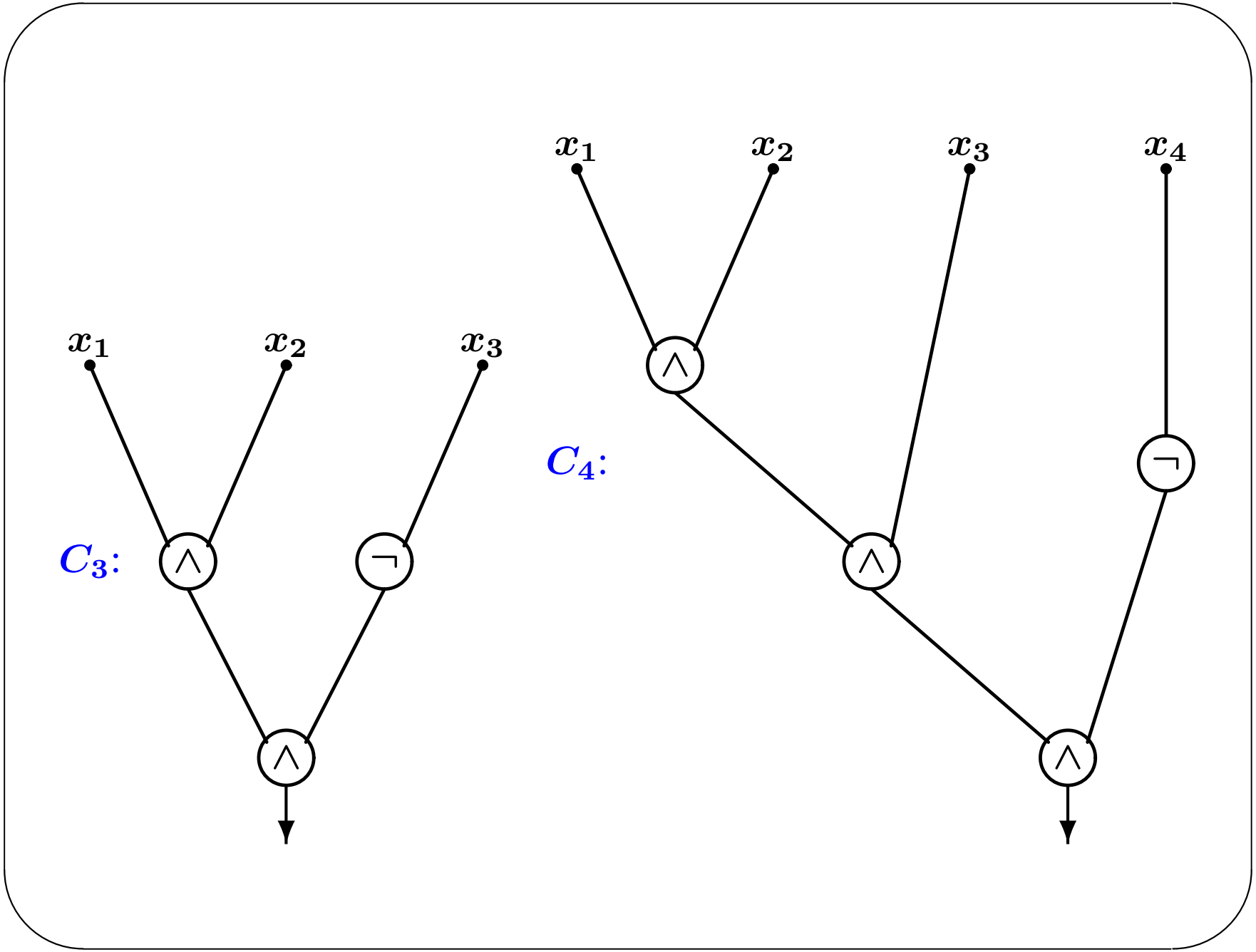
of circuits such that C_n has n inputs and computes the value of f for inputs of size n (or equivalently, inputs of size $\leq n$).

another example:

$$g(N) = \begin{cases} 1 & \text{if } N = 2^n - 2, \text{ for some } n, \\ 0 & \text{otherwise.} \end{cases}$$

Note that $g(N) = 1$ iff binary expansion of $N = 11 \cdots 110$

Here are examples of circuits of a sequence $C_1, C_2, \ldots, C_n, \ldots$ computing g


x_1
x_2
x_3
C_3:
x_1
x_2
x_3
x_4
C_4:

Uniform Circuits

A sequence

$$C_1, C_2, \ldots, C_n, \ldots$$

of circuits (C_n has n inputs) is uniform if there is an algorithm to construct each circuit C_n. Moreover, this algorithm runs in time polynomial in the size of the circuit C_n.

Note. There is a non–uniform sequence $C_1, C_2, \ldots, C_n, \ldots$ such that each C_n has size $O(n)$ and this sequence computes a non–computable function (like, Halting Problem).

Complexity Classes

For a decision problem $f : \{0, 1, 2, 3, \dots\} \longrightarrow \{0, 1\}$ the nth slice of f is the Boolean function

$$f_n : \{0, 1\}^n \longrightarrow \{0, 1\}$$

such that if $a_1 a_2 \cdots a_n$ is the binary expansion of integer N then $f_n(a_1, \dots, a_n) = f(N)$.

The class **P** (polynomial–time)

A decision problem f is in the class **P** iff there is a fixed polynomial $p(x)$ and a uniform sequence of circuits $C_1, C_2, \dots, C_n, \dots$ such that C_n computes f_n, the nth slice of f, and size$(C_n) \leq p(n)$.

class **P** = efficient problems

Any problem which has an efficient solution on any classical model of computation (such as, Turing machines, random access machines, etc.) is in class **P**, and vice versa.

Example of problems in **P**:

— Let S be the set of 0–1 square matrices. Then the function $f : S \longrightarrow \{0, 1\}$ defined as

$$f(M) = \det(M) \bmod 2$$

is in **P**.

The class **NP** (non-deterministic polynomial time)

$f : \{0, 1, 2, 3, \dots\} \longrightarrow \{0, 1\}$ a decision problem

$f_n : \{0, 1\}^n \longrightarrow \{0, 1\}$ its nth slice

f is in **NP** iff there is a uniform sequence

$$C_1, C_2, \dots, C_n, \dots$$

and a polynomial $p(x)$ such that:

(i) size$(C_n) \leq p(n)$;

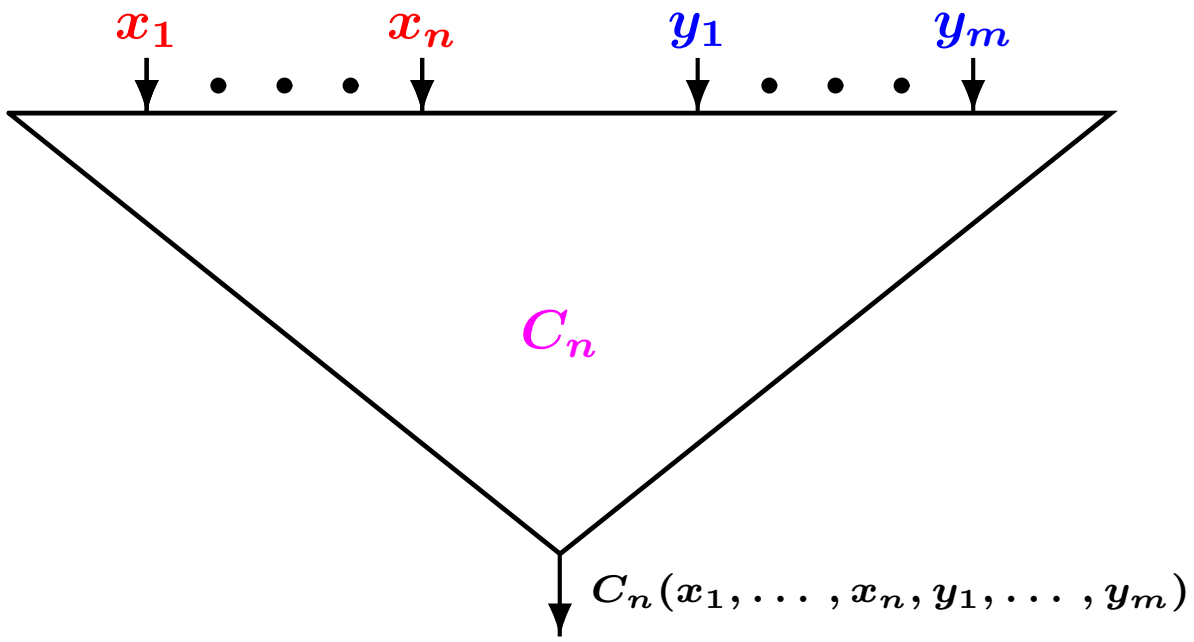
(ii) C_n has two distinct sets of inputs

$$x_1, \dots, x_n, \qquad y_1, \dots, y_m \qquad (m \leq p(n))$$

and we denote the output of C_n by $C_n(x_1, \dots, x_n, y_1, \dots, y_m)$; then

if $f_n(a_1, \ldots, a_n) = 1$ then there is a set of values $(b_1, \ldots, b_m)$ for $(y_1, \ldots, y_m)$ such that $C(a_1, \ldots, a_n, b_1, \ldots, b_m) = 1$;

If $f_n(a_1, \ldots, a_n) = 0$ then for every values $(d_1, \ldots, d_m)$ for $(y_1, \ldots, y_m)$ we have $C(a_1, \ldots, a_n, d_1, \ldots, d_m) = 0$.



Example: (Factorization Problem)

$$h\colon \{0,1,2,3,\dots\} \longrightarrow \{0,1\}$$

$$h(N) = \begin{cases} 1 & \text{if } N \text{ is a composite number,} \\ 0 & \text{otherwise.} \end{cases}$$

We show h is in **NP**. First we consider the function

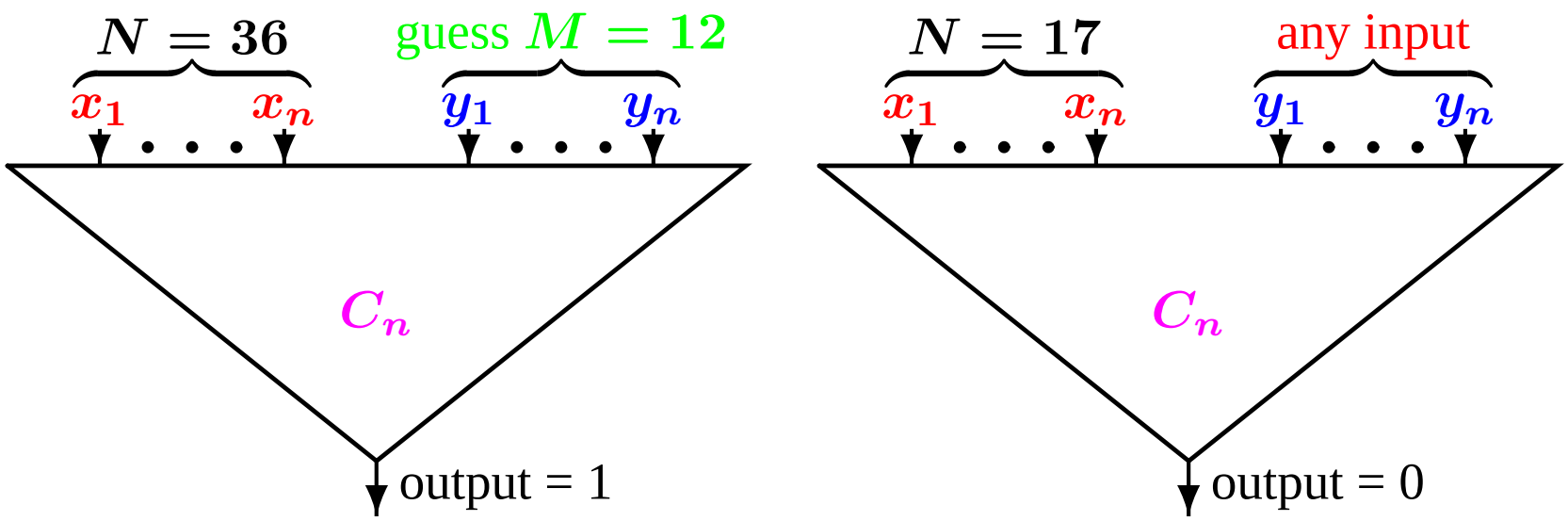
$$g(N,M) = \begin{cases} 1 & \text{if } M \leq N \text{ and } M \text{ divides } N, \\ 0 & \text{otherwise.} \end{cases}$$

Obviously g is in the class **P**. So there is a sequence of polynomial size circuits

$$C_1, C_2, \dots, C_n, \dots$$

that computes g, and C_n has two set of inputs: $x_1, \dots, x_n$ for N and $y_1, \dots, y_n$ for M.

Then the same sequence of circuits shows that h is in **NP**.



Equivalent characterization:

A decision problem $f(N)$ is in **NP** iff there is a decision problem $g(N, M)$ in **P** and a polynomial $p(x)$ such that for any input N of size n:

$$f(N) = 1 \quad \text{iff} \quad \exists M \left[\text{size}(M) \leq p(n) \text{ and } g(N, M) = 1\right]$$

$$f(N) = 0 \quad \text{iff} \quad \forall M \left[\text{if size}(M) \leq p(n) \text{ then } g(N, M) = 0\right]$$

Exercises:

Show that the following problems are in **NP**:

1. **Satisfiability:** A circuit is satisfiable if for some value of its input, the output is 1. The value of the function $S(C)$ is 1 iff the circuit C (given through some natural encoding of the circuits as 0–1 strings) is satisfiable.

2. **3–coloring of graphs:** The value of the function $X(G)$ is 1 iff the graph G can be colored with 3 colors (i.e, no two adjacent vertices have the same color).

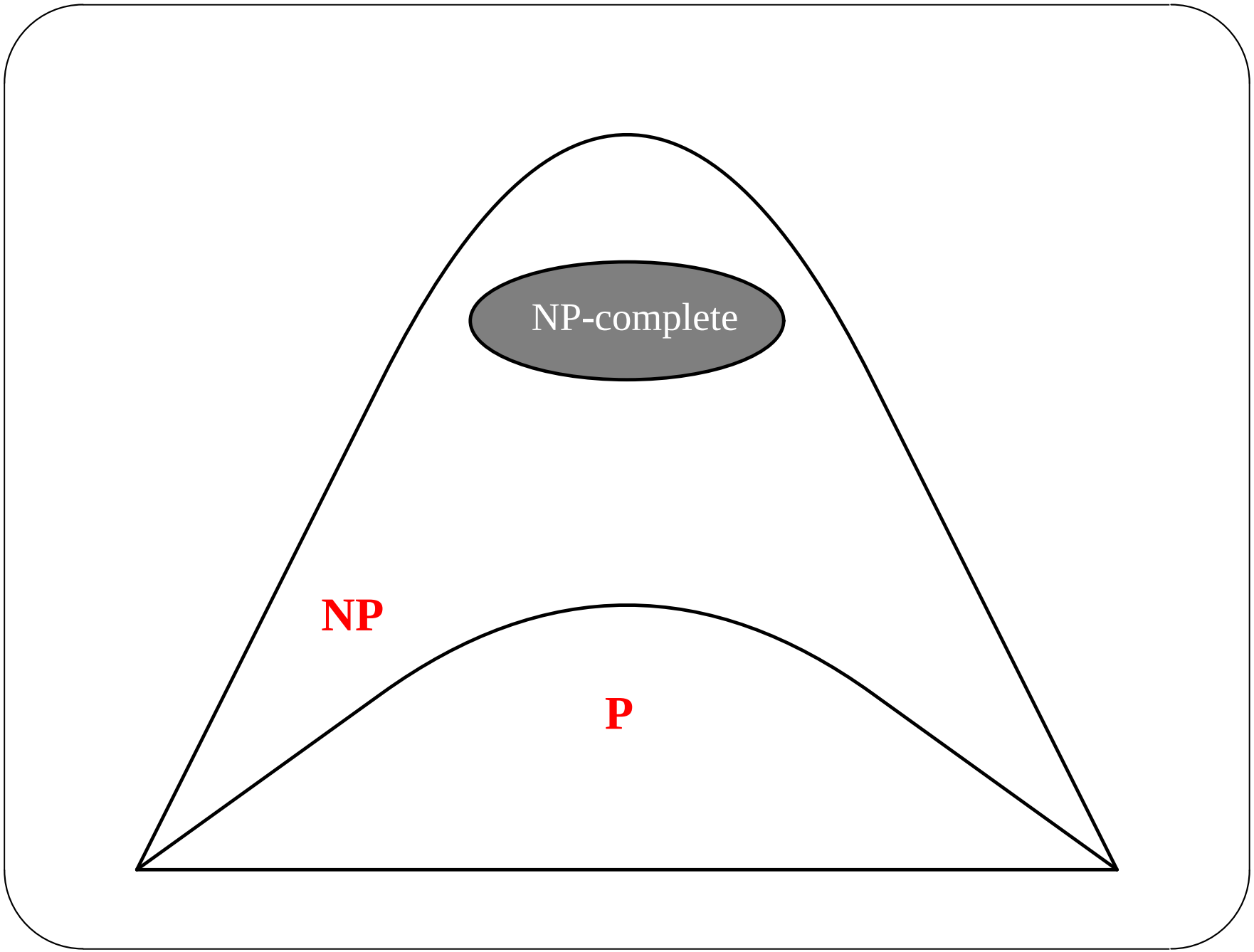
NP-complete Problems

A decision problem in **NP** is called **NP**–complete if any other problem in **NP** can be efficiently reduced to this problem. So an **NP**–complete problem is hardest problem in the class **NP** and an efficient solution for that implies efficient solution for any other problem in **NP**.

Formal definition: The function $f(N)$ (for $N \in \mathcal{A}$) defines a decision problem in **NP**. Then this problem is **NP**–complete if for any other problem $g(M)$ (for $M \in \mathcal{B}$) in **NP** there is a polynomial time computable function $\varphi\colon \mathcal{B} \longrightarrow \mathcal{A}$ such that

$$g(M) = 1 \quad \text{iff} \quad f(\varphi(M)) = 1.$$

Satisfiability and 3–coloring are **NP**–complete.


NP-complete
NP
P

Qubits

A single qubit is a two–state system, such as a two–level atom

We denote two orthogonal states of a single qubit as

$$\{\, |0\rangle \,, |1\rangle \,\}$$

So, the general stat of a single qubit can be represented as

$$a\,|0\rangle + b\,|1\rangle\,, \qquad \text{where } a, b \in \mathbb{C} \text{ and } |a|^2 + |b|^2 = 1.$$

A system of two qubits x and y:

product state: $(a_x\,|0\rangle_x + b_x\,|1\rangle_x) \otimes (a_y\,|0\rangle_y + b_y\,|1\rangle_y)$

entangled state: $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$

System of n qubits

Here we use a Hilbert space of dimension 2^n

The 2^n mutually orthogonal states of n qubits can be written as

$$\left\{ |x\rangle : x \text{ is an } n\text{–bit binary number} \right\}$$

for example, the orthogonal states of two qubits are

$$\left\{ |00\rangle , |01\rangle , |10\rangle , |11\rangle \right\}$$

the general state of such system is a superposition of the basic states

$$\sum_{j=1}^{2^n} \alpha_j |\psi_j\rangle , \qquad \sum_{j=1}^{2^n} |\alpha_j|^2 = 1, \qquad \psi_j \in \{0,1\}^n$$

Unitary Operations

An $m \times m$ matrix M over $\mathbb{C}$ is unitary if

$$M^\dagger \cdot M = I,$$

where $M^\dagger$ is the adjoint matrix of M; i.e., $M^\dagger = (M^{\mathrm{tr}})^*$, where * stands for complex conjugate.

$\mathbf{U}(m)$: the group of $m \times m$ unitary matrices

general form of a matrix in $\mathbf{U}(2)$

$$\begin{pmatrix} e^{i\alpha}\cos\theta & e^{i\beta}\sin\theta \\ -e^{-i\beta}\sin\theta & e^{-i\alpha}\cos\theta \end{pmatrix}$$

Quantum Gates

A unitary operation on n qubits is called an n–bit quantum gate

This gate is represented by a matrix in $\mathbf{U}(\mathbf{2}^{n})$

$\boldsymbol{\sigma_x}$, NOT gate, 1–bit gate

$$\begin{aligned} |0\rangle &\mapsto |1\rangle \\ |1\rangle &\mapsto |0\rangle \end{aligned} \qquad \boldsymbol{\sigma_x}(|a\rangle) = |1 \oplus a\rangle \qquad \boldsymbol{\sigma_x} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

$\boldsymbol{\sigma_z}$, phase change gate, 1–bit gate

$$\begin{aligned} |0\rangle &\mapsto |0\rangle \\ |1\rangle &\mapsto -|1\rangle \end{aligned} \qquad \boldsymbol{\sigma_z}(|a\rangle) = (-1)^a |a\rangle \qquad \boldsymbol{\sigma_z} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

$\boldsymbol{H}$, Hadamard transformation, 1–bit gate

$$\begin{aligned} |0\rangle &\mapsto \tfrac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ |1\rangle &\mapsto \tfrac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \end{aligned} \qquad \boldsymbol{H} = \tfrac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Controlled–NOT $=$ CNOT $= \boldsymbol{\Lambda_1(\sigma_x)}$, 2–bit gate

$$|a\rangle\,|b\rangle \mapsto \begin{cases} |a\rangle\,|b\rangle & \text{if } a = 0 \\ |a\rangle\,\sigma_x(|b\rangle) & \text{if } a = 1 \end{cases} \qquad \Lambda_1(\sigma_x)(|a\rangle\,|b\rangle) = |a\rangle\,|a \oplus b\rangle$$

$$\begin{aligned} |00\rangle &\mapsto |00\rangle\,, \\ |01\rangle &\mapsto |01\rangle\,, \\ |10\rangle &\mapsto |11\rangle\,, \\ |11\rangle &\mapsto |10\rangle\,. \end{aligned} \qquad \Lambda_1(\sigma_x) = \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & & 1 \\ & & 1 & \end{pmatrix}$$

Controlled–$U = \Lambda(U)$

an $(m+1)$–bit gate, where U is an m–bit gate

$$|a\rangle\,|b\rangle \mapsto \begin{cases} |a\rangle\,|b\rangle & \text{if } a = 0, \\ |a\rangle\,U(|b\rangle) & \text{if } a = 1. \end{cases} \qquad \Lambda(U) = \left(\begin{array}{c|c} I & \\ \hline & U \end{array}\right)$$

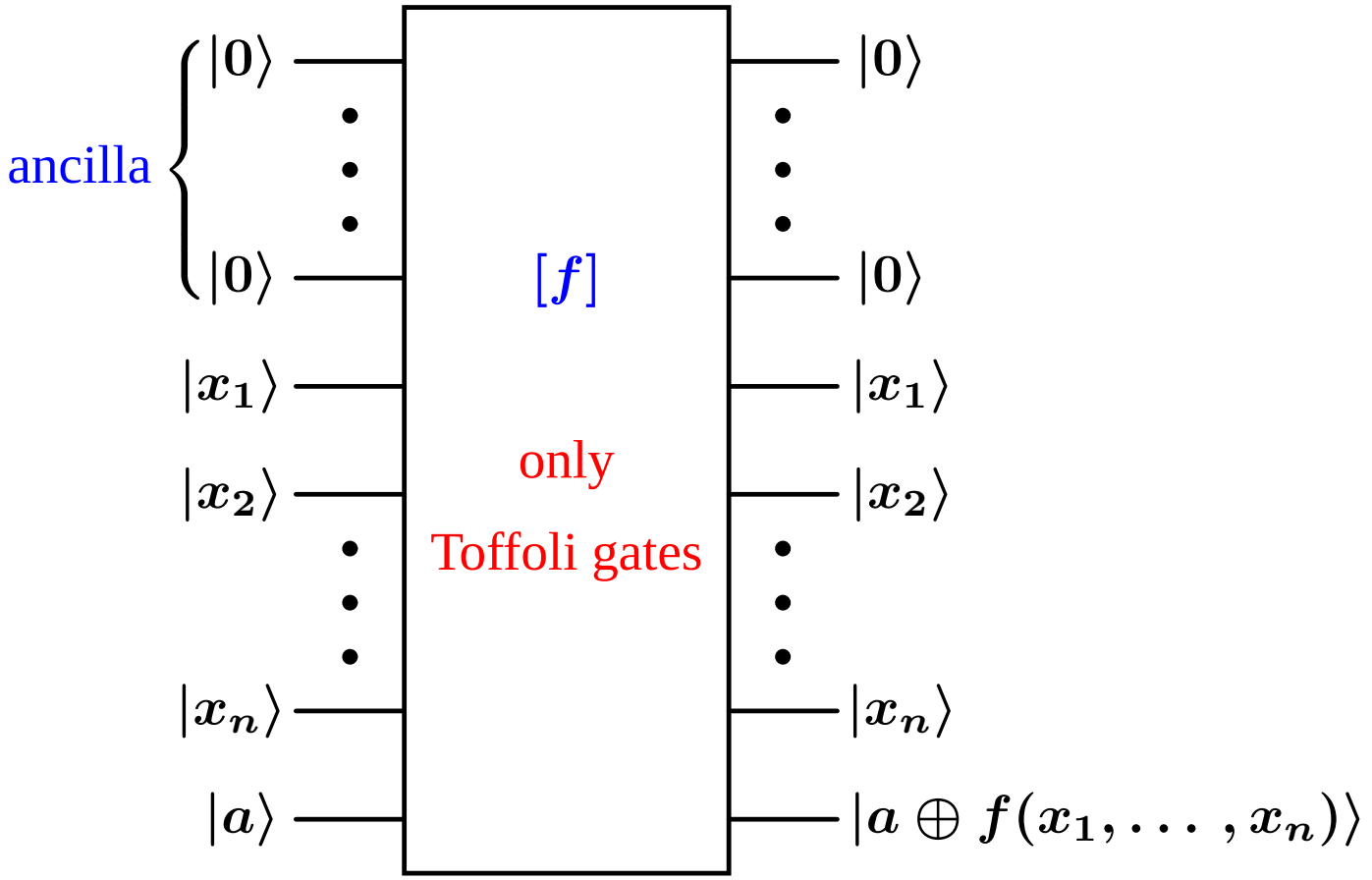
Tofolli gate, Controlled–Controlled–NOT, $\Lambda_2(\sigma_x) = \Lambda_1(\Lambda_1(\sigma_x))$,
3–bit gate

$$|a\rangle\,|b\rangle\,|c\rangle \mapsto \begin{cases} |a\rangle\,|b\rangle\,|c\rangle & \text{if } a = 0 \text{ or } b = 0, \\ |a\rangle\,|b\rangle\,\sigma_x(|c\rangle) & \text{if } a = b = 1. \end{cases}$$

$$\Lambda_2(\sigma_x)(|a\rangle\,|b\rangle\,|c\rangle) = |a\rangle\,|b\rangle\,|c \oplus (a \cdot b)\rangle$$

Reversible computing of Boolean functions

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



It can be shown that if the Boolean function $f : \{0,1\}^n \longrightarrow \{0,1\}$ can be computed by a Boolean circuit of size t, then the unitary transform

$$|x_1 \cdots x_n\rangle\,|b\rangle \mapsto |x_1 \cdots x_n\rangle\,|b \oplus f(x_1, \ldots, x_n)\rangle$$

can be performed by a quantum circuit of t^c (for some constant c) Toffoli gates.

n–bit Hadamard transform: $H^{\otimes n} = H \otimes H \otimes \cdots \otimes H$

perform the Hadamard transform H on each bit of n qubits $|x_1 \cdots x_n\rangle$

for example, in the case of $n = 2$ the result for the state $|01\rangle$ is

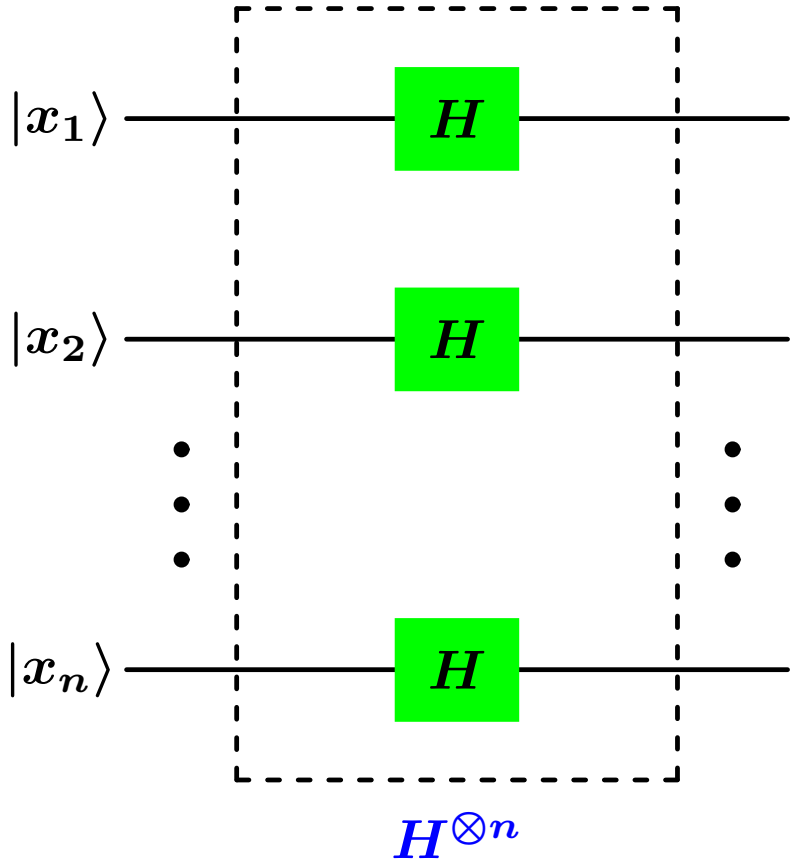
$$H(|0\rangle) \otimes H(|1\rangle) = \left[\tfrac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\right] \otimes \left[\tfrac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\right]$$

$$= \tfrac{1}{2}(|00\rangle - |01\rangle + |10\rangle - |11\rangle).$$

In general, for any 0–1 vector x of length n, the result of applying the Hadamard transform on $|x\rangle$ is equal to

$$H^{\otimes n}(|x\rangle) = \frac{1}{\sqrt{2^n}} \sum_{y \in \{0,1\}^n} (-1)^{x \cdot y} |y\rangle ,$$

where $x \cdot y$ is the inner product mod 2.


$|x_1\rangle$
H
$|x_2\rangle$
H
$|x_n\rangle$
H
$H^{\otimes n}$

Special Case: performing the Hadamard transform on n qubits $|0\cdots 0\rangle$ results in

$$\frac{1}{2^n}\sum_{y\in\{0,1\}^n}|y\rangle$$

So, starting from $|0\cdots 0\rangle$, by performing only n quantum gates it is possible to get the superposition of all 2^n basic states.

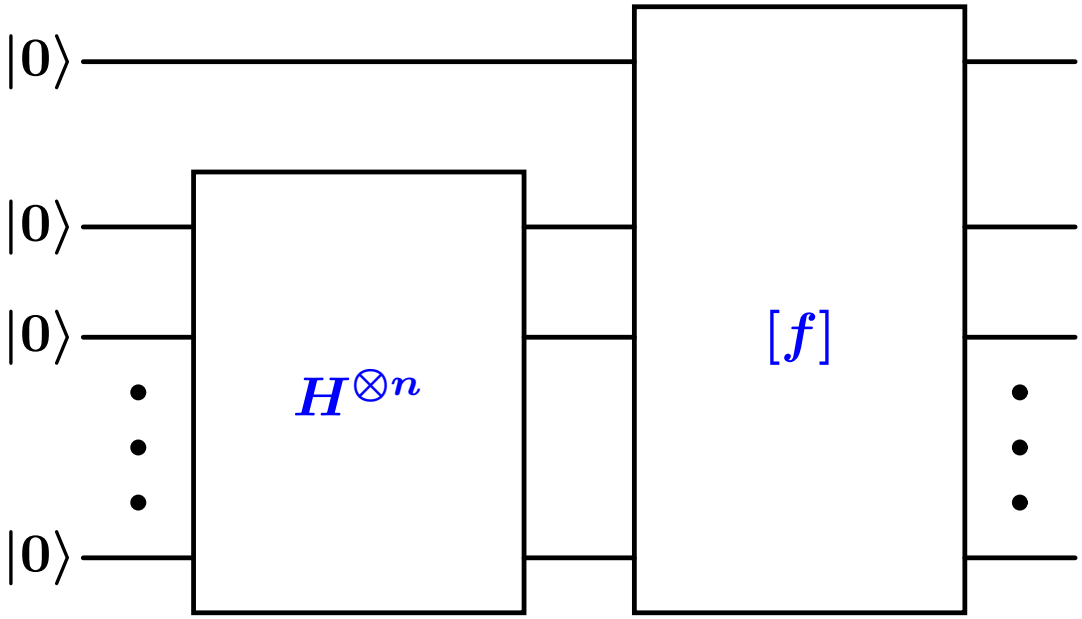
$$|00\rangle \xrightarrow{H^{\otimes 2}} \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)$$

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

$$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \xrightarrow{H^{\otimes 2}} \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

$$\frac{1}{\sqrt{2}}(|00\rangle - |11\rangle) \xrightarrow{H^{\otimes 2}} \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle)$$

interference: phase matters



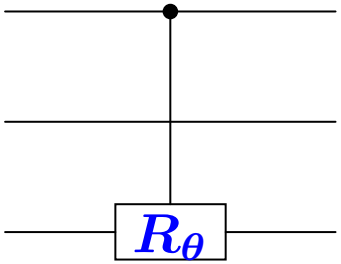
$$\text{output} = \frac{1}{\sqrt{2^n}} \sum_{a_1,\ldots,a_n \in \{0,1\}} |a_1 \cdots a_n\rangle \, |f(a_1, \ldots, a_n)\rangle$$

Universal Quantum Basis

A set $\mathcal{B}$ of quantum gates is called universal if every unitary operator $U \in \mathrm{U}(2^m)$, for any m, can be approximated within a given $\epsilon > 0$ with a quantum circuit consists of $\mathcal{B}$ gates.

Known Universal Quantum basis

Deutsch's gate: 3–bit gate



$$R_\theta = \begin{pmatrix} \cos\theta & i\sin\theta \\ i\sin\theta & \cos\theta \end{pmatrix}$$

$$\Lambda_2(R_\theta) = \left(\begin{array}{ccc|c} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \\ \hline & & & R_\theta \end{array}\right), \quad \theta \text{ is an irrational multiple of } \pi$$

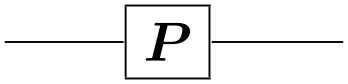
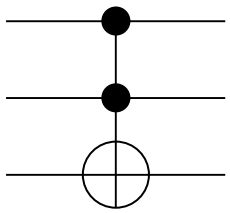
—Barenco: $\mathbf{\Lambda_1(R_\theta)}$, 2–bit gate

$$\mathbf{\Lambda_1(R_\theta)} = \left(\begin{array}{cc|c} \mathbf{1} & & \\ & \mathbf{1} & \\ \hline & & \boldsymbol{R_\theta} \end{array}\right)$$

— Basis consists of CNOT and all 1–bit unitary operations; i.e.,

CNOT and $\mathbf{U(2)}$

Shor's basis

Hadamard gate, H	$\frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$	
$P = \sigma_z{}^{\frac{1}{2}}$	$\begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$	
Toffoli gate	$\Lambda_2(\sigma_x)$	

Shor's basis is both universal and fault–tolerant

Proof of universality:

— Shor

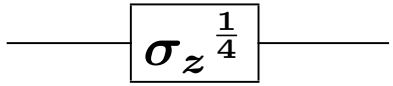
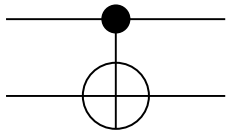
— Kitaev

— Boykin, Mor, Pulver, Roychowdhury, and Vatan

— Aharonov and Ben–Or

Another universal and fault–tolerant basis

by Boykin, Mor, Pulver, Roychowdhury, and Vatan

Hadamard gate, H	$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$	
$\sigma_z{}^{\frac{1}{4}}$	$\begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}$	
CNOT	$\Lambda_1(\sigma_x)$	

Measurement

Example 1:

$$|S\rangle = \sqrt{\frac{1}{3}}\,|00\rangle + \sqrt{\frac{2}{3}}\,|11\rangle$$

$$\text{measurement} \longrightarrow \begin{cases} |00\rangle & \text{with probability } \frac{1}{3} \\ |11\rangle & \text{with probability } \frac{2}{3} \end{cases}$$

Example 2: partial measurement

$$\frac{1}{\sqrt{2}}(|01\rangle - |10\rangle)$$

first system measured $\longrightarrow$

$$\begin{cases} |0\rangle & \text{with probability } \frac{1}{2} \text{ and the system collapses to } |1\rangle \\ |1\rangle & \text{with probability } \frac{1}{2} \text{ and the system collapses to } |0\rangle \end{cases}$$

Example 3:

$$\frac{1}{2}\,|0\rangle_1\,(|0\rangle_2 - |1\rangle_2) + \frac{1}{2}\,|1\rangle_1\,(|0\rangle_2 + |1\rangle_2)$$

first system measured $\longrightarrow$

$$\begin{cases} |0\rangle & \text{the system collapses to } \frac{1}{\|\sqrt{2}}(|0\rangle_2 - |1\rangle_2) \\ |1\rangle & \text{the system collapses to } \frac{1}{\sqrt{2}}(|0\rangle_2 + |1\rangle_2) \end{cases}$$

Computing with Quantum Circuits

C a quantum circuit with one of its qubits marked as output

The probability of C outputs 0 or 1 is the probability of measuring 0 or 1 at the output; i.e., if the state of the output is

$$|0\rangle_{\text{output}} |A_0\rangle + |1\rangle_{\text{output}} |A_1\rangle$$

then

$$\text{Prob}\,[C \text{ outputs } 0] = \| \, |A_0\rangle \, \|, \quad \text{Prob}\,[C \text{ outputs } 1] = \| \, |A_1\rangle \, \|,$$

where $\|V\|^2$ is the length of the vector $V \in \mathbb{C}^{2^m}$. Note that $\| \, |A_0\rangle \, \|^2 + \| \, |A_1\rangle \, \|^2 = 1$, because the transformation of C is unitary.

The quantum circuit C computes a Boolean function $f : \{0,1\}^n \longrightarrow \{0,1\}$ iff

Prob [on any input, the output of C is equal to the value of f] $\geq \frac{2}{3}$.

If in the above, the probability is always 1, then we say that C exactly computes f.

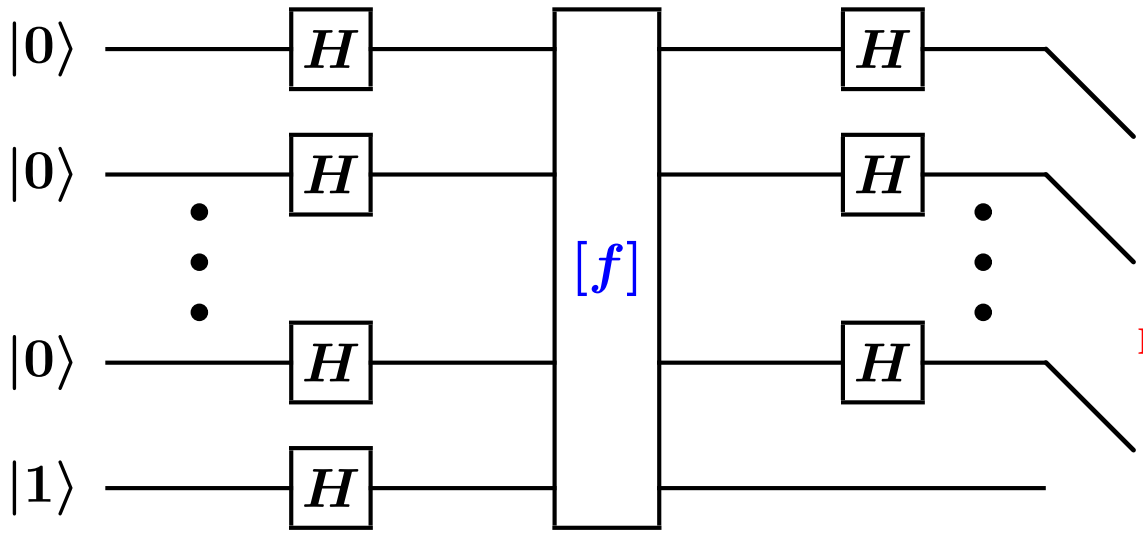
BQP is the class of problems computable by polynomial size uniform quantum circuit.

EQP is the class of problems exactly computable by polynomial size uniform quantum circuit.

Deutsch–Josza Algorithm

A Boolean function $f : \{0,1\}^n \longrightarrow \{0,1\}$ is given.

It is known that f is either constant or balanced (i.e., takes values 0 and 1 an equal number of times). The problem is to decide whether f is constant or balanced.



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

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Step 1: prepare the state

$$|00\cdots0\rangle\,(|0\rangle - |1\rangle)\,,$$

where the first register consists of n qubits.

Step 2: perform the Hadamard transform on the first n qubits

$$\sum_{x\in\{0,1\}^n} |x\rangle\,(|0\rangle - |1\rangle).$$

$$= \sum_{x\in\{0,1\}^n} |x\rangle\,|0\rangle - \sum_{x\in\{0,1\}^n} |x\rangle\,|1\rangle\,.$$

Step 3: apply

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

the result is

$$\sum_{x\in\{0,1\}^n} |x\rangle\,|0 \oplus f(x)\rangle - \sum_{x\in\{0,1\}^n} |x\rangle\,|1 \oplus f(x)\rangle\,.$$

note that for $a \in \{0,1\}$

$$|0 \oplus a\rangle - |1 \oplus a\rangle = (-1)^a(|0\rangle - |1\rangle).$$

so the result of this step is

$$\left[\sum_{x\in\{0,1\}^n} (-1)^{f(x)}\,|x\rangle\right](|0\rangle - |1\rangle).$$

note that

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

$$= \begin{cases} |A\rangle = \pm \sum_{x \in \{0,1\}^n} |x\rangle & \text{if } f \text{ is constant,} \\ |B\rangle & \text{if } f \text{ is balanced,} \end{cases}$$

where $|B\rangle$ is orthogonal to $|A\rangle$.

Step 4: apply the Hadamard transform $H^{\otimes n}$ on the first n qubits

note that

$$H^{\otimes n}\,|A\rangle = \pm\,|00\cdots0\rangle\,, \qquad \text{because } H^{-1} = H.$$

$H^{\otimes n}\,|B\rangle$ is orthogonal to $H^{\otimes n}\,|A\rangle = \pm\,|00\cdots0\rangle$, because the unitary operation $H^{\otimes n}$ maps orthogonal states to orthogonal ones.

$$\text{output} = \begin{cases} \pm\,|00\cdots0\rangle\,(|0\rangle - |1\rangle) & \text{if } f \text{ is constant} \\ |C\rangle\,(|0\rangle - |1\rangle) & \text{if } f \text{ is balanced} \end{cases}$$

where $|C\rangle$ is orthogonal to $|00\cdots0\rangle$

To get the answer, measure the first n qubits then

f is constant if and only if we observe n zeros

Simon's algorithm

$f : \{0,1\}^n \longrightarrow \{0,1\}^m$, with $m \geq n$,

there is a unique vector $s \in \{0,1\}^n$ such that for every $x, y \in \{0,1\}^n$ if $x \neq y$ then

$$f(x) = f(y) \Longleftrightarrow y = x \oplus s$$

Note that if $s = 0$ then f is a one–to–one function. In the case of $s \neq 0$, the set $\{0,1\}^n$ is partitioned to pairs of the form $\{x, x \oplus s\}$, where f is constant on each such pairs and it has different values for different pairs.

Simon's Problem

Instance: such function $f : \{0,1\}^n \longrightarrow \{0,1\}^m$ is given where $m \geq n$ and s may be $\mathbf{0}$ or not.

Question: What is s?

Simon's Subroutine

1. Prepare a state $|00\cdots0\rangle$ of $n+m$ zeros and apply the Hadamard transform $H^{\otimes n}$ on the first n qubits, producing the superposition

$$2^{-n/2} \sum_{x\in\{0,1\}^n} |x\rangle\,|0\rangle\,.$$

2. Compute $f(x)$ and save the result in the last qubit, producing

$$2^{-n/2} \sum_{x\in\{0,1\}^n} |x\rangle\,|f(x)\rangle\,.$$

3. Perform $H^{\otimes n}$ on the first n qubits again, producing the superposition

$$2^{-n} \sum_{y\in\{0,1\}^n} \sum_{x\in\{0,1\}^n} (-1)^{x\cdot y}\,|y\rangle\,|f(x)\rangle\,.$$

If $s \neq 0$ then $|y\rangle\,|f(x)\rangle = |y\rangle\,|f(x \oplus s)\rangle$

The amplitude of the state $|y\rangle\,|f(x)\rangle$ is equal to

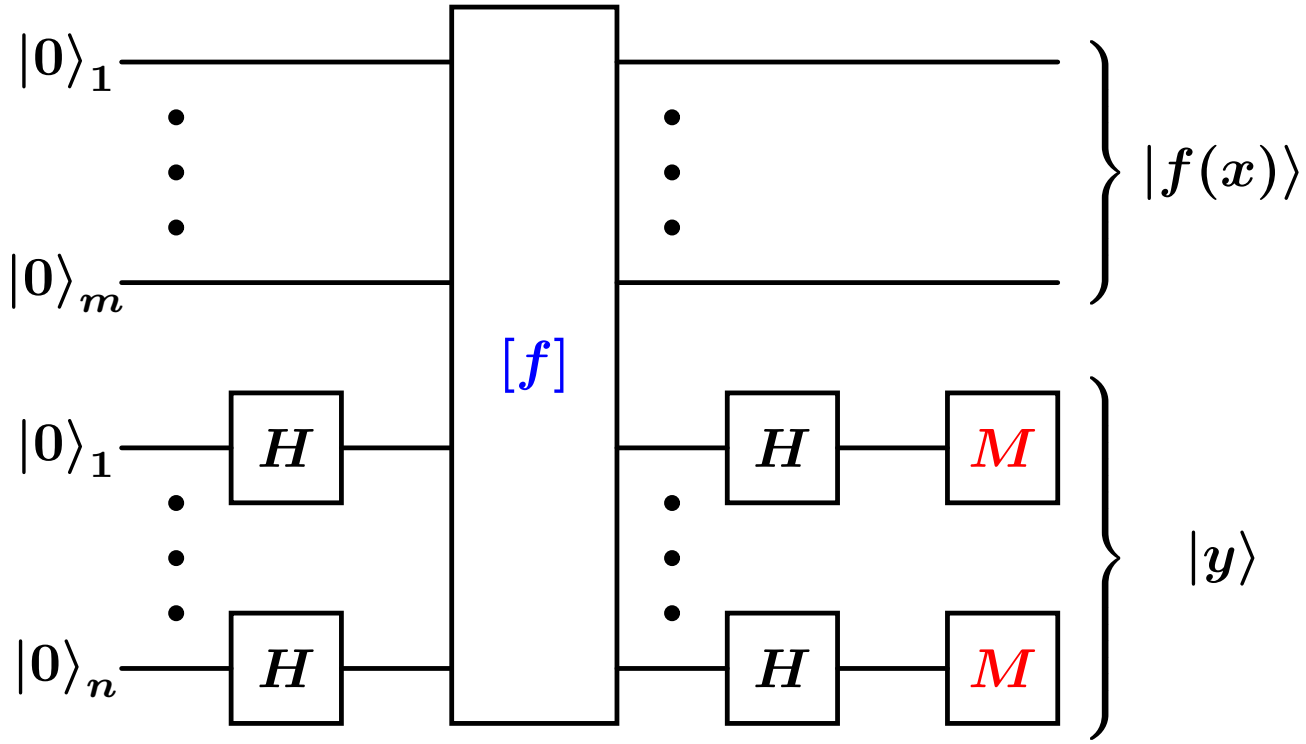
$$\alpha(x,y) = 2^{-n}\left((-1)^{x\cdot y} + (-1)^{(x\oplus s)\cdot y}\right) = \begin{cases} 2^{-n+1} & \text{if } y\cdot s = 0, \\ 0 & \text{otherwise.} \end{cases}$$

$$s^{\perp} = \{\, v \in \{0,1\}^n : s \cdot v = 0 \pmod 2 \,\}$$

The output superposition is actually is equal to

$$2^{-n+1} \sum_{y \in s^{\perp}} \sum_{x \in \{0,1\}^n} (-1)^{x\cdot y}\,|y\rangle\,|f(x)\rangle\,.$$

Therefore, measuring the first n qubits in this case is equivalent to randomly choosing an element of the $(n-1)$–dimensional space $s^{\perp}$ under the uniform distribution.


$|0\rangle_1$
$|0\rangle_m$
$|0\rangle_1$
$|0\rangle_n$
H
H
$[f]$
H
H
M
M
$|f(x)\rangle$
$|y\rangle$

Now after repeating the Simon's Subroutine $O(n)$ times and measuring the first n qubits, with high probability we find a basis for $s^{\perp}$; and thus it is possible to find out s.

Actually the following lemma shows that we need to repeat the subroutine at most $\mathbf{5}n$ times to make sure that the probability of failure is at most $\mathbf{2}^{-\lambda n}$, for some fixed $\lambda > 0$.

Lemma. Let X be an ℓ–dimensional subspace of $\{0,1\}^n$. We choose randomly and independently 5ℓ vectors from X under the uniform distribution. The probability that there are less than ℓ independent vectors among the chosen ones is at most $\mathbf{2}^{-\lambda \ell}$, for some fixed $\lambda > 0$.

Grover's algorithm: Quantum search

A Boolean function $T: \{0,1\}^n \longrightarrow \{0,1\}$ such that $T(x) = 1$ for only one value $x = x_0$.

Find: x_0.

We assume that T can be evaluated in unit time.

Any classical algorithm (deterministic or probabilistic) for this problem needs to check at least $\frac{1}{2}2^n$ vectors of $\{0,1\}^n$.

Grover's algorithm shows the quantum computer can solves this problem in time $O(\sqrt{2^n})$.

Fix $c \in \{0,1\}^n$, we define the unitary operator S_c as

$$\text{for } |x\rangle \in \mathbb{C}^{2^n}, \quad S_c(|x\rangle) = \begin{cases} -|x\rangle & \text{if } x = c, \\ |x\rangle & \text{otherwise.} \end{cases}$$

We will apply S_0, for $c = 0$, and S_{x_0}. The operator S_0 can be computed efficiently (in time polynomial in n);

In the beginning we do not know x_0, but still it is possible to compute S_{x_0} efficiently, if we assume that $T(x)$ can be computed efficiently on each given input $x \in \{0,1\}^n$.

The algorithm is based on iteration of the following transform:

$$G = -H^{\otimes n} S_0 H^{\otimes n} S_{x_0},$$

First we prepare the uniform superposition

$$|A\rangle = 2^{-n/2} \sum_{x \in \{0,1\}^n} |x\rangle$$

Then we apply the transform G on $|A\rangle$ repeatedly

The operator G maps each superposition of the form

$$k\,|x_0\rangle + \ell \sum_{x \neq x_0} |x\rangle$$

to itself.

Initially

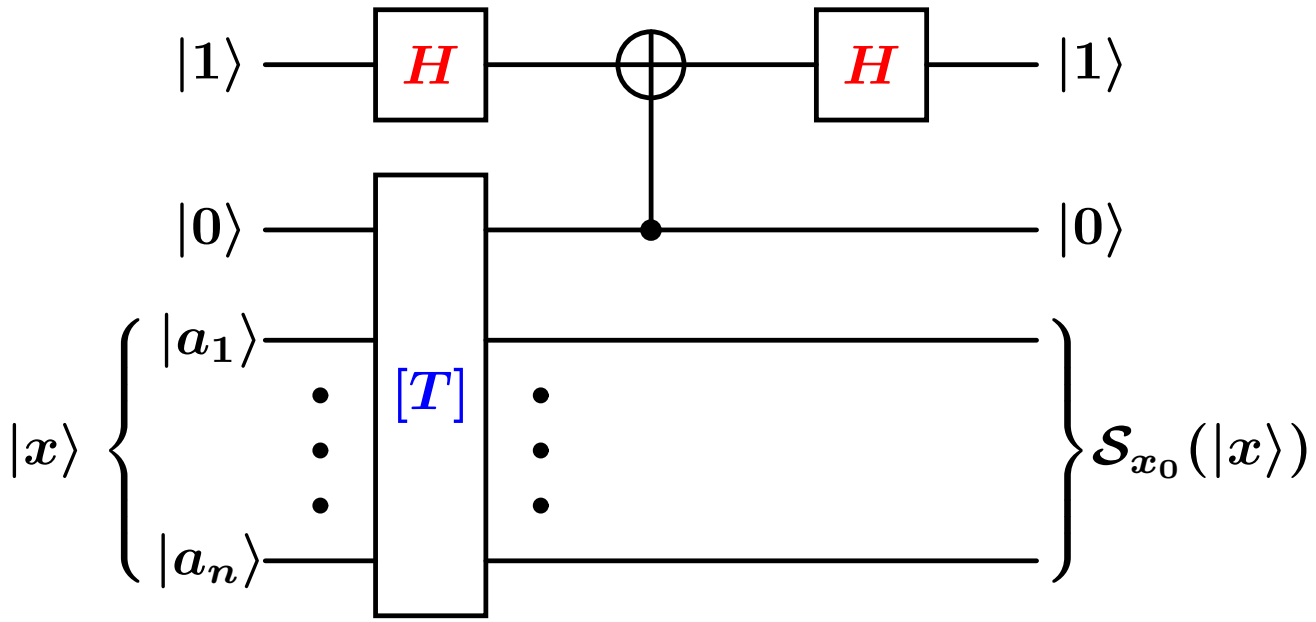
$$k_0 = \ell_0 = \frac{1}{\sqrt{2^n}}$$

and after j iteration of G, for $\sin\theta = \frac{1}{\sqrt{2^n}}$,

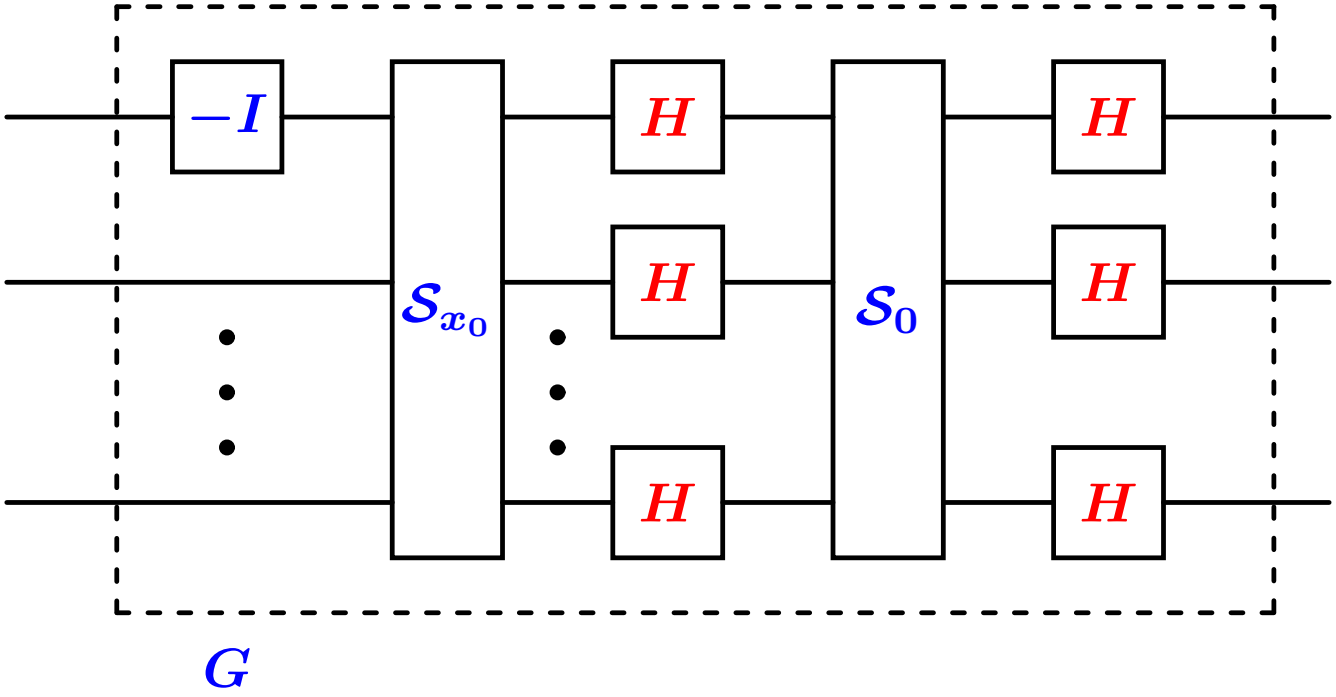
$$\begin{aligned} k_j &= \sin((2j+1)\theta), \\ \ell_j &= \frac{1}{\sqrt{2^n - 1}} \cos((2j+1)\theta), \end{aligned}$$

After m iterations the probability of measuring x_0,i.e. ${k_m}^2$, is very close to 1 if

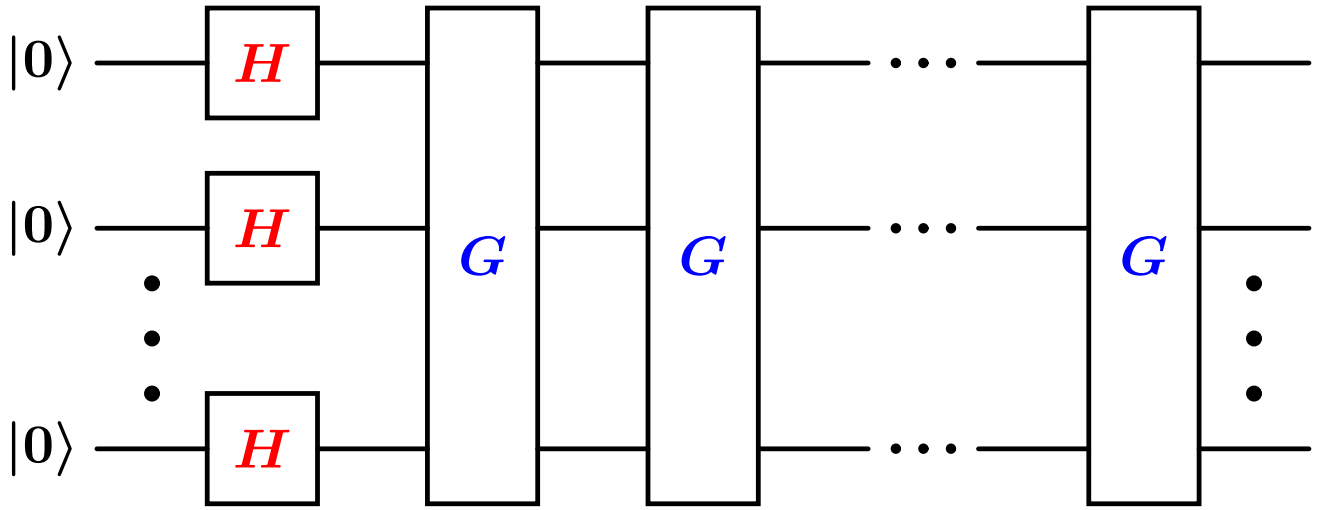
$$m = \lfloor \frac{\pi}{4}\sqrt{2^n} \rfloor$$



Computing the unitary operator $\mathcal{S}_{x_0}$



Computing the unitary operator G



Search algorithm

Quantum Factorization Algorithm

input: composite number N.

output: a factor of N.

To solve this problem, it is enough to find a nontrivial solution for

$$x^2 \equiv 1 \pmod{N}$$

i.e., $x \equiv a \pmod{N}$, $|a| < N$, and $a \neq \pm 1$.

Then N divides $(a+1)(a-1)$ but N does not divide $a \pm 1$, so

$$\text{g.c.d.}(N, a+1) \text{ or } \text{g.c.d.}(N, a-1)$$

is a factor of N.

Example: If $N = 15$ then $a = 11$ is a nontrivial solution, because $11^2 = 121 = 8 \times 15 + 1 \equiv 1 \pmod{15}$; then $(a+1)(a-1) = 120 = 8 \times 15$ and

$$\text{g.c.d.}(15, 12) = 3 \text{ and g.c.d.}(15, 10) = 5.$$

How to find a nontrivial solution?

Given N choose a random $y < N$.

If y and N are coprime, then let r be the order of y mod N; i.e., the least positive integer r such that

$$y^r \equiv 1 \pmod{N}.$$

For example if $N = 15$ and $y = 8$ then the order of y is 4, because $8^4 \equiv 1 \pmod{15}$ and 4 is the least number with this property.

If r is even then $x = y^{r/2}$ is a candidate for a nontrivial solution, because for random y

$$\text{Prob}\,[(N, y) = 1] > \frac{1}{\log N};$$

and if $(N, y) = 1$

$$\text{Prob}\left[r \text{ is even and } y^{r/2} \not\equiv \pm 1 \pmod{N}\right] > \frac{1}{2}.$$

So if we have an algorithm which provides the order r of $y \pmod N$ (when $(N, y) = 1$) then we can find a factor of N. Thus the original factorization problem reduces to the following.

input: integers x and N such that $(x, N) = 1$.

output: least r such that $x^r \equiv 1 \pmod N$.

Shor's Algorithm

Let $q = 2^{\ell}$ such that $N^2 \leq q \leq 2N^2$. Each register is of ℓ qubits.

Step (I): using Hadamard transform, needs $O(\ell) = O(\log N)$ operations

$$|0\rangle\,|0\rangle \xrightarrow{H^{\otimes \ell}} \sum_{a=0}^{q-1} |a\rangle\,|0\rangle$$

Step (II): needs $O(\log^2 N)$ operations

$$\sum_{a=0}^{q-1} |a\rangle\,|0\rangle \longmapsto \sum_{a=0}^{q-1} |a\rangle\,|x^a \pmod N\rangle$$

Step (III): apply Fourier transform, needs $O(\log^2 n)$ operations

$$\sum_{a=0}^{q-1} |a\rangle \, |x^a \pmod N\rangle \longmapsto \sum_{a=0}^{q-1} \sum_{c=0}^{q-1} e^{\frac{2\pi i a c}{q}} \, |c\rangle \, |x^a \pmod N\rangle$$

Step (IV): observe the machine

$$|c\rangle \, |x^a \pmod N\rangle$$

Step (V): there is at most one fraction $\frac{d}{r}$ with $0 < r < N$ such that $(d, r) = 1$ and $\left|\frac{c}{q} - \frac{d}{r}\right| \leq \frac{1}{2q}$. Find $\frac{d}{r}$. This needs poly$(\log n)$ operations. **output:** r

$$\text{Prob}\,[r \text{ is the order of } x \pmod N] \geq \frac{c}{\log \log N}.$$

Quantum Fourier Transform

Fix integer $q > 0$.

Standard basis for $\mathbb{C}^q$: $|0\rangle, |1\rangle, \ldots, |q-1\rangle$

$\mathcal{F}_q$: the Fourier transform on $\mathbb{C}^q$

$$|a\rangle \xrightarrow{\mathcal{F}_q} \frac{1}{\sqrt{q}} \sum_{c=0}^{q-1} e^{\frac{2\pi i a c}{q}} |c\rangle$$

We construct a quantum circuit for F_q when $q = 2^\ell$. In this case the state $|a\rangle$ is also represented by binary expansion of a as $|a_{\ell-1} a_{\ell-2} \cdots a_0\rangle$.

Example: $q = 4$, the standard basis for $\mathbb{C}^4$ is

$$|0\rangle = |00\rangle\,, \quad |1\rangle = |01\rangle\,, \quad |2\rangle = |10\rangle\,, \quad |3\rangle = |11\rangle$$

$$\begin{aligned}
|00\rangle &\xrightarrow{\mathcal{F}_4} \frac{1}{2}\left(|00\rangle + |01\rangle + |10\rangle + |11\rangle\right) \\
|01\rangle &\xrightarrow{\mathcal{F}_4} \frac{1}{2}\left(|00\rangle + e^{i\frac{\pi}{2}}\,|01\rangle + e^{i\pi}\,|10\rangle + e^{i\frac{3\pi}{2}}\,|11\rangle\right) \\
|10\rangle &\xrightarrow{\mathcal{F}_4} \frac{1}{2}\left(|00\rangle + e^{i\pi}\,|01\rangle + |10\rangle + e^{i\pi}\,|11\rangle\right) \\
|11\rangle &\xrightarrow{\mathcal{F}_4} \frac{1}{2}\left(|00\rangle + e^{i\frac{3\pi}{2}}\,|01\rangle + e^{i\pi}\,|10\rangle + e^{i\frac{\pi}{2}}\,|11\rangle\right)
\end{aligned}$$

Gates:

— H_j: applies the Hadamard on jth bit

— $S_{j,k}$ applies the following transform on bits in positions j and k with $j < k$

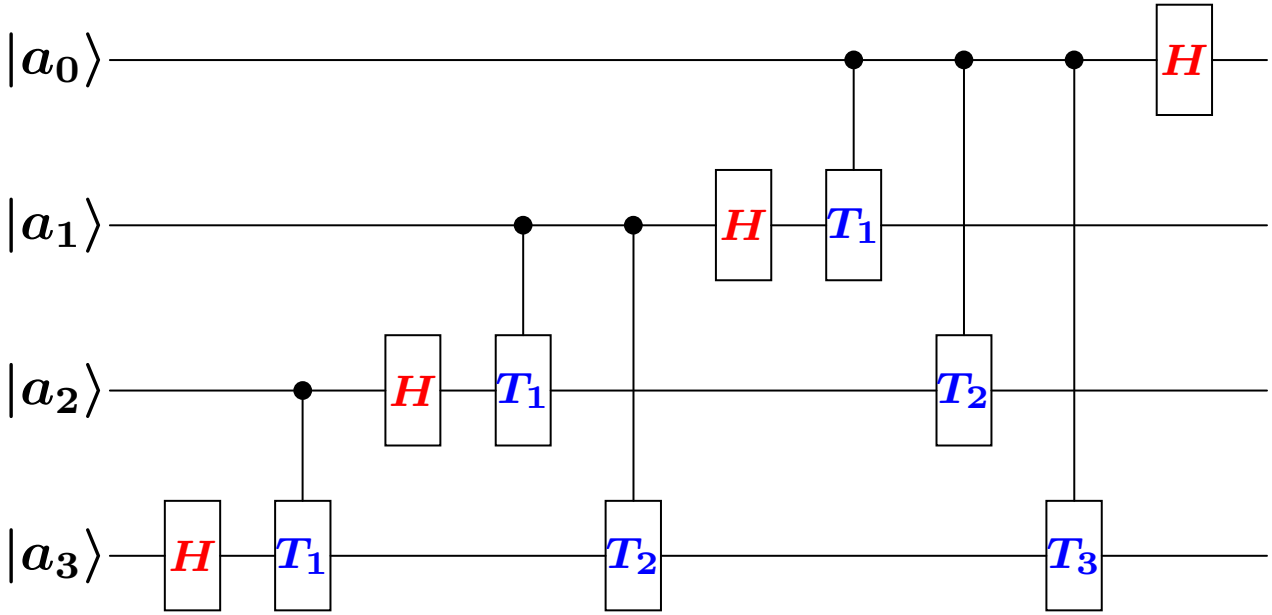
$$S_{j,k} = \Lambda_1(T_{k-j}) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\theta_{k-j}} \end{pmatrix}, \qquad \theta_{k-j} = \pi/2^{k-j}$$

apply from left to right $H_{\ell-1}$, $S_{\ell-2,\ell-1}$, $H_{\ell-2}$, $S_{\ell-3,\ell-1}$, $S_{\ell-3,\ell-2}$, $H_{\ell-3}, \cdots H_1$, $S_{0,\ell-1}$, $S_{0,\ell-2}, \cdots S_{0,2}$, $S_{0,1}$, H_0

For example, when $\ell = 4$ the order of the gates is

$$H_3, S_{2,3}, H_2, S_{1,3}, S_{1,2}, H_1, S_{0,3}, S_{0,2}, S_{0,1}, H_0$$

The number of gates is $\ell(\ell-1)/2$.



Computing the Fourier transform $\mathcal{F}_4$

Kitaev's Algorithm

Given coprime integers x and n, find r the order of x mod n.

$$\mathcal{U}: \ |a\rangle \mapsto |ax \ (\text{mod } n)\rangle \ , \qquad a = 0, 1, \ldots , n-1$$

The unitary operator $\mathcal{U}$ is a permutation on $\{|0\rangle , |1\rangle , \ldots , |n-1\rangle\}$.
Each nontrivial cycle of this permutation has length r:

$$|a\rangle \longrightarrow |ax\rangle \longrightarrow \left|ax^2\right\rangle \longrightarrow \cdots \longrightarrow \left|ax^{r-1}\right\rangle \longrightarrow |ax^r\rangle = |a\rangle$$

Corresponding to each cycle, we can find r eigenvectors of $\mathcal{U}$:

$$|x_\lambda\rangle = \frac{1}{\sqrt{r}} \left(|a\rangle + \lambda \, |ax\rangle + \lambda^2 \left|ax^2\right\rangle + \cdots + \lambda^{r-1} \left|ax^{r-1}\right\rangle\right)$$

$$\mathcal{U} \, |x_\lambda\rangle = \lambda^{-1} \, |x_\lambda\rangle \, , \qquad \text{if } \lambda = e^{\frac{2i\pi k}{r}} , \ k = 0, 1, \ldots , r-1$$

We will consider the eigenvectors when $a = 1$; these are the following vectors, for $k = 0, 1, \ldots, r-1$

$$|\psi_k\rangle = \frac{1}{\sqrt{r}} \sum_{j=0}^{r-1} \lambda_k^{-j} \left|x^j \pmod n\right\rangle = \frac{1}{\sqrt{r}} \sum_{j=0}^{r-1} e^{2\pi i \frac{jk}{r}} \left|x^j \pmod n\right\rangle$$

and they satisfy the following relation

$$|1\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |\psi_k\rangle$$

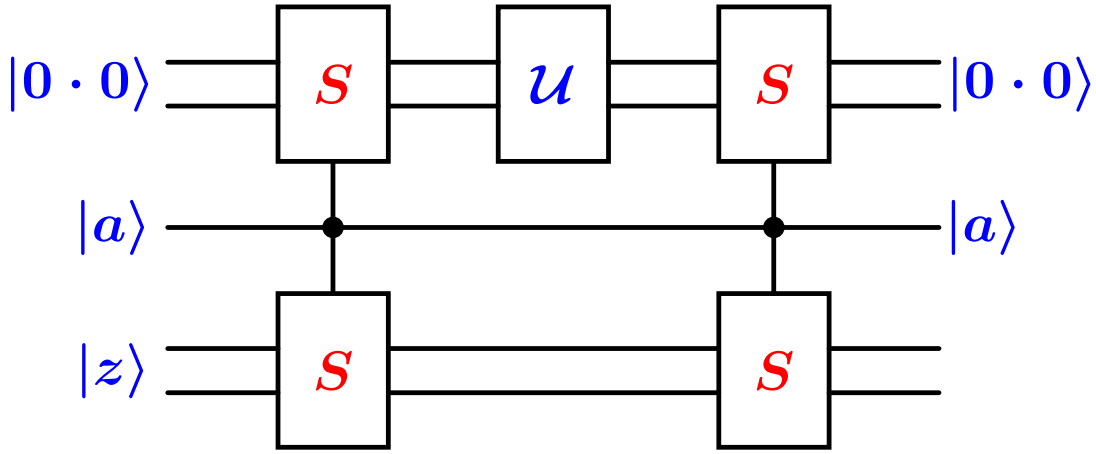
Although we do not have an efficient algorithm to prepare any of eigenvectors $|\psi_k\rangle$, for the moment we assume we have access to these eigenvectors and we show how we can compute the corresponding eigenvalues.

So we start with a unitary operator $\mathcal{U}$ on $\mathbb{C}^n$ and an eigenvector $|y_\lambda\rangle$ of $\mathcal{U}$ corresponding to the eigenvalue $\lambda = e^{2\pi i\phi}$, where ϕ is a real number and $0 \leq \phi < 1$.

The goal is to design a quantum algorithm that computes (with high probability) ϕ, and consequently the eigenvalue λ.

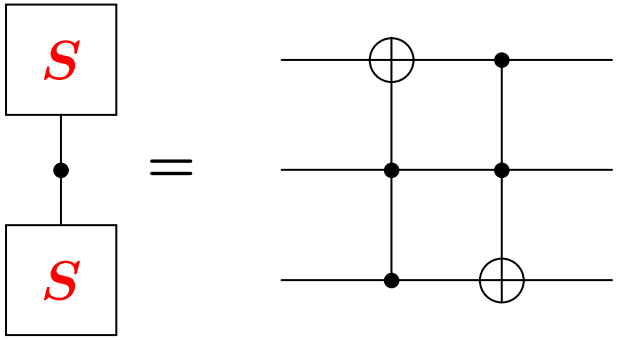
First, we define the operator $\Lambda(\mathcal{U})$ on $\mathbb{C}^{n+1}$as follows

$$\Lambda(\mathcal{U})\left(|a\rangle \otimes |z\rangle\right) = \begin{cases} |0\rangle \otimes |z\rangle & \text{if } a = 0, \\ |1\rangle \otimes \mathcal{U}(|z\rangle) & \text{if } a = 1. \end{cases}$$



$\Lambda(\mathcal{U})$ circuit, for $\mathcal{U}\,|0\cdots 0\rangle = |0\cdots 0\rangle$

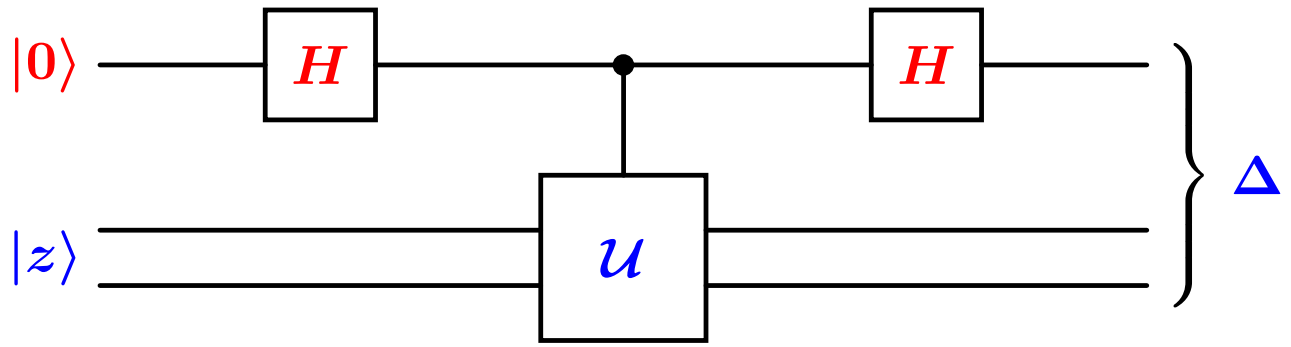
Controlled–swap gate:



An algorithm for finding the phase ϕ, given the eigenvector $|y_\lambda\rangle$

First, we define the unitary operator Δ on $\mathbb{C}^{n+1}$ as the composition of the following operators. On the input $|a\rangle\,|z\rangle$ (the first register is of one qubit and the second one is of ℓ qubits)

- apply the Hadamard transform H on the first register,
- apply $\Lambda(U)$,
- apply the Hadamard transform H again on the first register.



Then

$$\Delta(|0\rangle\,|y_\lambda\rangle) = |\xi_\lambda\rangle\,|y_\lambda\rangle$$

$$|\xi_\lambda\rangle = \tfrac{1+e^{2\pi i\phi}}{2}\,|0\rangle + \tfrac{1-e^{2\pi i\phi}}{2}\,|1\rangle$$

So measuring the first register results in 0 or 1 with the probabilities

$$p_0 = \frac{1+\cos 2\pi\phi}{2}, \qquad p_1 = \frac{1-\cos 2\pi\phi}{2}$$

We apply Δ with s different control qubits and measure the control qubits. Let the result of this measurements be $b_1, \ldots, b_s$. Let m be the number of the times we observe "1"; i.e., $m = b_1 + \cdots + b_s$. Since $b_1, \ldots, b_s$ are independent random variables, m/s is very close to p_1 with high probability. More specifically, for any constant $\delta > 0$,

$$\text{Prob}\left[\left|\frac{m}{s} - p_1\right| > \delta\right] \le e^{-c_\delta s}$$

Thus it is possible, by applying s times the operator Δ, to find $\cos 2\pi\phi$ with precision δ and error probability $\leq \varepsilon = \exp(-c_\delta s)$ (so $s = O(\log(1/\varepsilon))$).

By applying the same procedure on $i\mathcal{U}$ instead of $\mathcal{U}$, we will find $-\sin 2\pi\phi$. So the phase ϕ can be measured with precision δ and error probability $\leq \varepsilon$ by applying $O(\log(1/\varepsilon))$ times $\Lambda(\mathcal{U})$.

How to computes the order r?

Since all the eigenvalues of $\mathcal{U}$, are of the form $\exp\left(2\pi i \frac{k}{r}\right)$, it is enough to compute one of the phases $\phi = k/r$.

Note that different phases are at least $1/r$ apart each other, so if we compute any phase with $\log(r) \leq \log(n)$ bit precision, we get the exact value of r.

We can not prepare any eigenvector of $\mathcal{U}$ efficiently, the state $|1\rangle$ is an equally weighted superposition of r eigenvectors of $\mathcal{U}$:

$$|1\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |\psi_k\rangle$$

So if we apply the above procedure to the state $|\mathbf{1}\rangle$ with t control qubits (for an appropriate value of t), the result is the superposition

$$\frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} \underbrace{|\xi_{\lambda_k}\rangle \cdots |\xi_{\lambda_k}\rangle}_{t \text{ times}} |\psi_k\rangle \,.$$

The state $|\xi_{\lambda_k}\rangle \cdots |\xi_{\lambda_k}\rangle$ of control qubits can be written as a superposition $\sum_x \alpha_x |x\rangle$, where each x is a t–bit integer. The terms in the last superposition can be grouped as

$$\sum_{c \in A_k} \alpha_c |c\rangle + \sum_{d \in B_k} \alpha_d |d\rangle \,,$$

where A_k is the set of all integers c such that the ratio of the numbers of 1's in the binary expansion of c gives the correct value p_1 corresponding to the eigenvalue λ_k. Then

$$\sum_{c \in A_k} |\alpha_c|^2 \geq 1 - \varepsilon.$$

Then the measurement of the control qubits gives the value k/r (for k randomly chosen from the integers $0, 1, \ldots, r-1$) with high probability. Therefore, the denominator of the result is the value of the order of x mod n.