

# Distribución pública de llaves y Factorización de enteros



**ALGORITMOS RSA y  
SHOR**

**II Escuela de Computación Cuántica**

**8-12/01/2024**



**Aldo Delgado Hidalgo**

# Introducción

El tema central de esta clase es la **factorización de números enteros como productos de números primos**. Esto es, dado un entero  $N$  escribimos éste como  $N = p_1 \cdot p_2 \cdots p_m$  donde los números  $p_1, p_2, \dots, p_m$  son primos, es decir, son números enteros divisibles sólo por 1 y por si mismos.

La factorización de números enteros como productos de números primos juega un rol central en el esquema RSA para la transmisión segura de información. Para quebrar la seguridad de este esquema es necesario factorizar eficientemente. Sin embargo, no existen algoritmos clásicos conocidos para esta tarea que sean eficientes.

Sin embargo, haciendo uso de un computador cuántico es posible desarrollar un algoritmo, el célebre algoritmo de Shor, que puede factorizar de forma eficiente.

# Contenidos

En esta clase examinaremos los siguientes temas:

## 1. Criptografía clásica básica

1. Nociones básicas de criptografía por medio de One-Time Pad
2. Sistema criptográfico RSA para distribución de llaves públicas

## 2. Factorización de enteros en productos de números primos

1. Transformada de Fourier Cuántica
2. Estimación de Fase
3. Búsqueda de Período
4. Algoritmo de Shor

# Criptografía clásica básica

# Criptografía clásica básica

En este capítulo vamos a introducir nociones básicas de criptografía. Usaremos para ello el esquema One-time Pad para la transmisión segura de información y luego estudiaremos el algoritmo RSA.

Nuestro objetivo principal es mostrar que la seguridad de este último descansa en que no se conocen algoritmos eficientes para factorizar.

# Criptografía clásica básica

## One-time pad

Consideremos la comunicación entre dos partes, típicamente llamadas Alice y Bob.

El mensaje  $m$  a comunicar corresponde a un texto en un cierto lenguaje, el cual puede ser representado digitalmente por medio de una secuencia binaria. Por ejemplo, cadenas de bits de largo  $n$ :

$$m = \underbrace{01001010\dots 0}_{n \text{ bits}}$$

Luego, el conjunto de todos los mensajes posibles corresponde al conjunto de todas las cadenas binarias de largo  $n$ , esto es

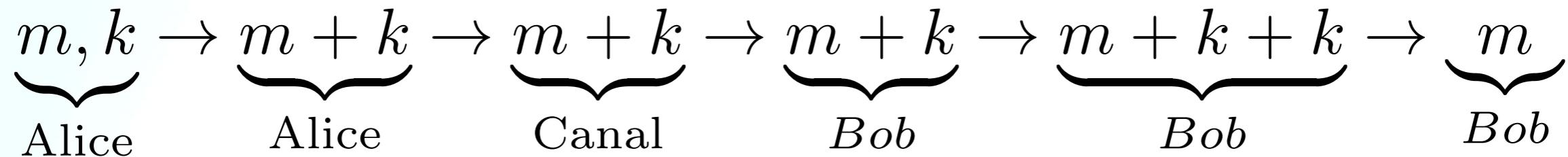
$$m \in \{0, 1\}^n$$

# Criptografía clásica básica

## One-time pad

El problema que enfrentan Alice y Bob es la transmisión segura de un mensaje entre ellos, es decir, de forma tal que sólo el destinatario definido previamente pueda tener acceso al mensaje.

La idea del One-time pad (Frank Miller, 1882) es la siguiente: se genera una llave  $k$ , la cual también es una secuencia de bits de largo  $n$ . Alice suma a su mensaje  $m$  la llave  $k$  por medio de adición binaria y transmite el resultado  $m + k$  a Bob, quien suma la llave  $k$  al mensaje transmitido para recuperar el mensaje, es decir, realiza la operación  $(m + k) + k$ . Debido a las propiedades de la adición binaria tenemos que  $(m + k) + k = m$ .



# Criptografía clásica básica

## One-time pad

Como vemos, en el One-time pad la llave  $k$  es conocida por Alice y Bob previamente. Esto supone implícitamente la existencia de una **canal seguro para distribuir** la llave entre Alice y Bob.

Imaginemos que existe una tercera parte, Eve, interesada en obtener el mensaje  $m + k$  y que de hecho, lo ha obtenido por algún medio.

Dado que la llave  $k$  es desconocida para Eve, ella debe probar aleatoriamente todas las llaves posibles, ósea  $m + k + k'$ . ¿Cuantas llaves posibles existen? Bien, si  $n = 2048$  entonces el conjunto de las llaves posibles son  $2^{2048}$ , es decir, aproximadamente  $10^{617}$  elementos.

# Criptografía clásica básica

## One-time pad

El super-computador más rápido (Frontier, USA Energy Department, 2023) puede efectuar  $10^{18}$  operaciones por segundo.

Supongamos que cada operación consiste en realizar  $m + k + k'$  y determinar si es una cadena admisible.

Entonces, si usamos este computador para probar sistemáticamente todas las llaves  $k'$ , nos tomará  $10^{599}$  segundos o, equivalentemente,  $10^{592}$  años.

# Criptografía clásica básica

## One-time pad

El método One-time Pad debe satisfacer algunas condiciones para garantizar la seguridad de la información transmitida:

1. La llave  $k$  debe ser tan larga como el mensaje  $m$ .
2. La llave  $k$  debe ser aleatoria, esto es, generada con probabilidad uniforme en el espacio de las llaves posibles.
3. La llave  $k$  no debe ser utilizada nuevamente.
4. La llave  $k$  debe mantenerse en secreto absoluto por Alice y Bob.

**¿Cómo garantizamos  
esto último?**

# Criptografía clásica básica

## Distribución pública de llaves

Ron Rivest, Adi Shamir y Leonard Adleman introdujeron en 1978 un nuevo algoritmo criptográfico para reemplazar el algoritmo del National Bureau of Standards. El algoritmo RSA introduce dos ideas muy importantes:

**Encriptación con llave pública:** se omite la necesidad de un correo para entregar las llaves por medio de un canal seguro antes de transmitir el mensaje.

En el algoritmo RSA existen llaves de encriptación y decriptación. Cada parte tiene su propia llave de encriptación y decriptación. La llaves de encriptación son públicas mientras que las de decriptación no lo son.

Las llaves son hechas de forma tal que las llaves de decriptación no se pueden deducir fácilmente de las llaves de encriptación.

# Criptografía clásica básica

## Distribución pública de llaves

Ron Rivest, Adi Shamir y Leonard Adleman introdujeron en 1978 un nuevo algoritmo criptográfico para reemplazar el algoritmo del National Bureau of Standards. El algoritmo RSA introduce dos ideas muy importantes:

**Firma digital:** el receptor puede necesitar verificar que el mensaje transmitido efectivamente proviene de quien lo envía (firma).

Esto es hecho usando la llave de decrptación de quien envía el mensaje y la firma puede ser verificada por cualquier parte usando la llave pública de encriptación.

Firmas no pueden ser falsificadas.

# Criptografía clásica básica

## Distribución pública de llaves

En un sistema de distribución de llave pública cada usuario o parte tiene sus propios procedimientos de encriptación y decrptación, los cuales denotamos por  $E$  y  $D$ , respectivamente. Estos procedimientos actúan sobre el mensaje  $m$ .

Los procedimientos  $E$  y  $D$  dependen de las llaves públicas y privadas y tienen las siguientes propiedades:

1.  $D(E(m)) = m$
2.  $E(D(m)) = m$
3.  $E$  y  $D$  son fáciles de calcular.
4. El que  $E$  sea público no compromete el secreto de  $D$ . Es decir, conocer  $E$  no permite conocer fácilmente  $D$

# Criptografía clásica básica

## Distribución pública de llaves

Ahora, Bob desea enviar un mensaje  $m$  a Alice. Para ello, Bob lee desde un archivo público  $E_A$ , el procedimiento de encriptación de Alice, y lo usa para crear el texto encriptado  $c = E_A(m)$ .

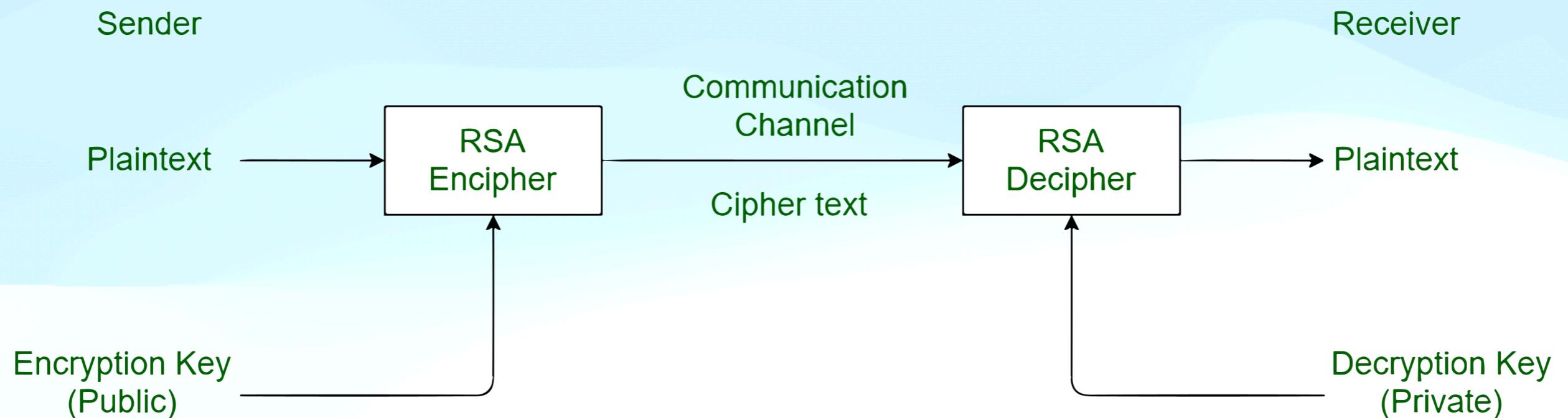
Luego, Alice recibe  $c = E_A(m)$  y aplica su procedimiento de deciptación  $D_A$  sobre  $c$  para obtener  $m = D_A(E_A(m))$ . Esto es algo que sólo Alice puede hacer.

Alice podría contestar a Bob enviándole otro mensaje  $m'$  por medio de leer en el archivo público el procedimiento de encriptación  $E_B$  de Bob.

Todo lo que necesitamos de los usuarios es su acuerdo para poner en el archivo público sus respectivos procedimientos de encriptación. Ninguna comunicación privada es necesaria de antemano.

# Criptografía clásica básica

## Distribución pública de llaves



# Criptografía clásica básica

## Distribución pública de llaves

Consideremos ahora un escenario donde sólo dos partes o usuarios se comunican. El mensaje  $m$  será un número entero positivo entre 0 y  $n - 1$ .

Las llaves de encriptación y decriptación serán los pares  $(e, n)$  y  $(d, n)$ , respectivamente, con  $e, d$  y  $n$  enteros positivos.

El procedimiento de encriptación será:

$$c = m^e \bmod(n)$$

El procedimiento de decriptación será:

$$m = c^d \bmod(n)$$

# Criptografía clásica básica

## Distribución pública de llaves

El problema de Alice es cómo elegir los enteros  $e, d$  y  $n$  de manera tal que el conocimiento de la llave pública  $(e, n)$  no permita calcular eficientemente la llave privada  $(d, n)$ .

Primero, asumiremos que  $n = p \cdot q$ , donde los números primos grandes y distantes  $p$  y  $q$  son elegidos aleatoriamente y mantenidos en secreto por Alice.

El entero  $n$  será público. Sin embargo, dado que no se conoce ningún algoritmo clásico eficiente para descomponer  $n$  como un producto de primos, los números  $p$  y  $q$  permanecen en secreto.

# Criptografía clásica básica

## Distribución pública de llaves

Ahora que hemos escogido  $n$  debemos determinar valores de  $e$  y  $d$  apropiados.

Escogemos  $d$  aleatoriamente como un entero grande. Además,  $d$  debe ser tal que

$$\gcd(d, (p - 1) \cdot (q - 1)) = 1$$

Esto es, los números  $d$  y  $(p - 1) \cdot (q - 1)$  deben tener como divisor solamente el número 1. En otras palabras, deben ser coprimos.

Recordemos que  $d$  es elegido por Alice y forma parte de la llave privada, es decir,  $d$  permanece en secreto.

# Criptografía clásica básica

## Distribución pública de llaves

Ahora que hemos escogido  $n$  y  $d$  debemos determinar el valor de  $e$  apropiado.

Escogemos  $e$  tal que sea el inverso multiplicativo de  $d$ . Se debe cumplir entonces

$$e \cdot d = 1 \bmod(\phi(n))$$

donde el valor de la función  $\phi(n)$  es igual al número de enteros positivos menores que  $n$  que son co-primos con  $n$ . Por las propiedades de esta función tenemos que:

$$\phi(n) = (p - 1) \cdot (q - 1)$$

y por teoría de número tenemos que  $e$  existe sólo si  $d$  y  $(p - 1) \cdot (q - 1)$  son co-primos.

# Criptografía clásica básica

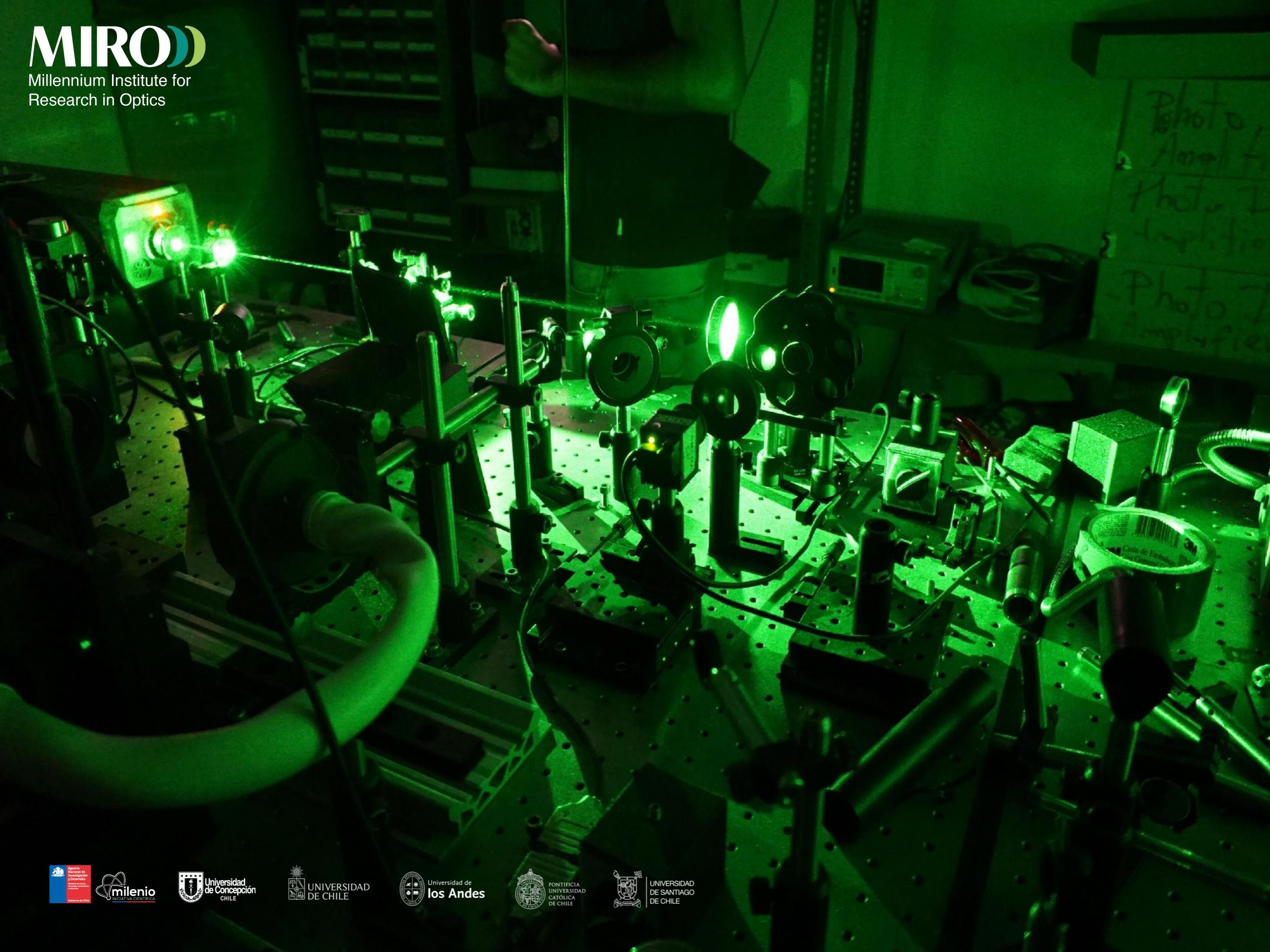
## Distribución pública de llaves

La ecuación

$$e \cdot d = 1 \bmod(\phi(n))$$

podría ser usada por Bob para obtener el valor de  $d$ , puesto que  $e$  es parte de la clave pública de la misma forma que  $n$ .

Sin embargo, el valor de la función  $\phi(n)$  se puede calcular eficientemente sólo si conocemos la descomposición de  $n$  en sus factores primos  $p$  y  $q$ . Éstos son conocidos por Alice pero no por Bob. En este hecho subyace la seguridad del protocolo de distribución de llaves públicas de RSA, puesto que no existen algoritmos conocidos que sean eficientes para descomponer un entero en sus factores primos.



# **Factorización de enteros en productos de números primos**

# Factorización...

El objetivo de este capítulo es introducir el algoritmo de Shor para la factorización eficiente de enteros en productos de números primos.

Para ello definiremos la Transformada de Fourier Cuántica y construiremos su circuito cuántico. Luego, la aplicaremos a los problema de estimación de fase y búsqueda de período, mostrando los respectivos circuitos cuánticos.

Finalmente, basándonos en la Transformada de Fourier Cuántica y su aplicación a la estimación de fase, enunciaremos el algoritmo de Shor.

# Factorización...

## Transformada de Fourier Cuántica

Consideremos un sistema cuántico cuyos estados están definidos por medio de una base **B** de dimension **d**.

Los elementos de la base

$$B = \{|x\rangle\} \text{ con } x = 0, \dots, d - 1$$

son mutuamente ortogonales y tienen norma unitaria, es decir,

$$\langle x|y\rangle = \delta_{x,y}$$

Cualquier estado cuántico (puro) se puede escribir como:

$$|\psi\rangle = \sum_{x=0}^{d-1} c_x |x\rangle \text{ con } c_x \in \mathbb{C} \text{ y } \sum_{x=0}^{d-1} |c_x|^2 = 1$$

# Factorización...

## Transformada de Fourier Cuántica

La acción de la Transformada de Fourier Cuántica (QFT) sobre los elementos de la base **B** es

$$U_{FT}|x\rangle = \frac{1}{\sqrt{d}} \sum_{y=0}^{d-1} \omega^{xy} |y\rangle \quad \forall x, y \in B,$$

donde la constante  $\omega$  es definida como

$$\omega = e^{\frac{2\pi i}{d}}.$$

Luego,  $\omega$  es una raíz compleja de la unidad tal que

$$\omega^{xd} = 1 \quad \forall x = 0, \dots, d-1.$$

# Factorización...

## Transformada de Fourier Cuántica

La acción de la Transformada de Fourier Cuántica (QFT) sobre un estado arbitrario

$$|\psi\rangle = \sum_{x=0}^{d-1} c_x |x\rangle$$

está dada por la expresión

$$\begin{aligned} U_{FT}|\psi\rangle &= \sum_{x=0}^{d-1} c_x [U_{FT}|x\rangle] \\ &= \sum_{x=0}^{d-1} c_x \frac{1}{\sqrt{d}} \sum_{y=0}^{d-1} \omega^{xy} |y\rangle \\ &= \sum_{y=0}^{d-1} \left[ \frac{1}{\sqrt{d}} \sum_{x=0}^{d-1} \omega^{xy} c_x \right] |y\rangle \end{aligned}$$

# Factorización...

## Transformada de Fourier Cuántica

Luego, la Transformada de Fourier Cuántica (QFT) transforma las amplitudes de probabilidad de acuerdo a la regla

$$c_x \rightarrow \frac{1}{\sqrt{d}} \sum_{x=0}^{d-1} \omega^{xy} c_x$$

tal como la Trasformada de Fourier discreta clásica.

# Factorización...

## Transformada de Fourier Cuántica

La Transformada de Fourier Cuántica es unitaria. Esto quiere decir que su transpuesta conjugada es su inversa

$$U_{FT}(U_{FT})^\dagger = \mathbb{I}$$

Verifiquemos esta propiedad.

Debemos mostrar que la acción del operador  $U_{FT}(U_{FT})^\dagger$  es la identidad.

Luego,  $U_{FT}(U_{FT})^\dagger$  transforma los elementos de la base B sobre si mismos. Notemos que:

$$(U_{FT})^\dagger |x\rangle = \frac{1}{\sqrt{d}} \sum_{y=0}^{d-1} \omega^{-xy} |y\rangle$$

# Factorización...

## Transformada de Fourier Cuántica

$$\begin{aligned} U_{FT}[(U_{FT})^\dagger |m\rangle] &= U_{FT}\left[\frac{1}{\sqrt{d}} \sum_{y'=0}^{d-1} \omega^{-my'} |y'\rangle\right] \\ &= \frac{1}{\sqrt{d}} \sum_{y'=0}^{d-1} \omega^{-my'} [U_{FT}|y'\rangle] \\ &= \frac{1}{\sqrt{d}} \sum_{y'=0}^{d-1} \omega^{-my'} \frac{1}{\sqrt{d}} \sum_{x=0}^{d-1} \omega^{y'x} |x\rangle \\ &= \frac{1}{d} \sum_{x=0}^{d-1} \sum_{y'=0}^{d-1} \omega^{(x-m)y'} |x\rangle = \frac{1}{d} \sum_{x=0}^{d-1} d\delta_{x,m} |x\rangle \\ &= |m\rangle \quad \forall m \in \mathbb{B} \end{aligned}$$

# Factorización...

## Transformada de Fourier Cuántica

Ahora:

1. Supondremos que la dimensión  $d$  es una potencia entera de 2, es decir,  $d = 2^n$ , o equivalentemente un sistema de  $n$  qubits, y
2. haremos uso de la representación binaria de los números enteros, esto es:

$$j = j_1 2^{n-1} + j_2 2^{n-2} + \dots + j_{n-1} 2 + j_n 2^0 = \sum_{m=1}^n j_m 2^{n-m}$$

donde

$$j_1, j_2, \dots, j_n \in \{0,1\}$$

Denotamos entonces:  $|j\rangle = |j_1 j_2 \dots j_n\rangle = |j_1\rangle \otimes |j_2\rangle \otimes \dots \otimes |j_n\rangle$

# Factorización...

## Transformada de Fourier Cuántica

Podemos escribir la acción de la transformada de Fourier como:

$$\begin{aligned} U_{FT}|x_0 \dots x_{n-1}\rangle &= \frac{1}{\sqrt{2^n}} \sum_{y=0}^{d-1} \omega^{xy} |y\rangle \\ &= \frac{1}{\sqrt{2^n}} \sum_{y_1=0,1} \dots \sum_{y_n=0,1} \omega^{x \sum_{m=1}^n y_m 2^{n-m}} |y_1 \dots y_n\rangle \\ &= \frac{1}{\sqrt{2^n}} \sum_{y_1=0,1} \dots \sum_{y_n=0,1} \prod_{m=1}^n \omega^{x y_m 2^{n-m}} |y_1 \dots y_n\rangle \\ &= \frac{1}{\sqrt{2^n}} \sum_{y_1=0,1} \dots \sum_{y_n=0,1} \prod_{m=1}^n e^{2\pi i x y_m 2^{-m}} |y_1 \dots y_n\rangle \end{aligned}$$

# Factorización...

## Transformada de Fourier Cuántica

$$\begin{aligned} U_{FT}|x_1 \dots x_n\rangle &= \frac{1}{\sqrt{2^n}} \sum_{y_1=0,1} \dots \sum_{y_n=0,1} \prod_{m=1}^n e^{2\pi i x y_m 2^{-m}} |y_m\rangle \\ &= \frac{1}{\sqrt{2^n}} \prod_{m=1}^n \sum_{y_m=0,1} e^{2\pi i x y_m 2^{-m}} |y_m\rangle \\ &= \frac{1}{\sqrt{2^n}} \prod_{m=1}^n (|0_m\rangle + e^{2\pi i x 2^{-m}} |1_m\rangle) \\ &= \frac{1}{\sqrt{2}} (|0_1\rangle + e^{2\pi i x 2^{-1}} |1_1\rangle) \otimes \frac{1}{\sqrt{2}} (|0_2\rangle + e^{2\pi i x 2^{-2}} |1_2\rangle) \otimes \dots \\ &\quad \otimes \frac{1}{\sqrt{2}} (|0_n\rangle + e^{2\pi i x 2^{-n}} |1_n\rangle) \end{aligned}$$

# Factorización...

## Transformada de Fourier Cuántica

Examinemos ahora el argumento de la exponencial que aparece en la estado de cada uno de los qubits. Este es de la forma

$$x2^{-l} = \left( \sum_{m=1}^n x_m 2^{n-m} \right) 2^{-l}$$

y lo podemos expandir como

$$\begin{aligned} x2^{-l} &= \sum_{m=1}^n x_m 2^{(n-l)-m} \\ &= \sum_{m=1}^{n-l} x_m 2^{(n-l)-m} + \sum_{m=n-l+1}^n x_m 2^{(n-l)-m} \\ &= \sum_{m=1}^{n-l} x_m 2^{(n-l)-m} + \sum_{k=1}^l x_{k+n-l} 2^{-k} \end{aligned}$$

# Factorización...

## Transformada de Fourier Cuántica

Luego, la exponencial resulta ser

$$e^{2\pi i x 2^{-l}} = e^{2\pi i \sum_{m=1}^{n-l} x_m 2^{(n-l)-m}} e^{2\pi i \sum_{k=1}^l x_{k+n-l} 2^{-k}}$$

El primer término a la derecha puede ser omitido por ser una potencia entera de 1, de modo que obtenemos

$$e^{2\pi i x 2^{-l}} = e^{2\pi i \sum_{k=1}^l x_{k+n-l} 2^{-k}}$$

# Factorización...

## Transformada de Fourier Cuántica

Así entonces el argumento de la exponencial resulta ser

$$\sum_{k=1}^1 x_{k+n-1} 2^{-k} = x_n 2^{-1}$$

$$\sum_{k=1}^2 x_{k+n-2} 2^{-k} = x_{n-1} 2^{-1} + x_n 2^{-2}$$

$$\sum_{k=1}^3 x_{k+n-3} 2^{-k} = x_{n-2} 2^{-1} + x_{n-1} 2^{-2} + x_n 2^{-3}$$

⋮

$$\sum_{k=1}^n x_{k+n-n} 2^{-k} = x_1 2^{-1} + x_2 2^{-2} + x_3 2^{-3} + \cdots + x_n 2^{-n}$$

# Factorización...

## Transformada de Fourier Cuántica

Así entonces el argumento de la exponencial resulta ser para cada qubit se una fase que se acumula progresivamente:

$$\sum_{k=1}^1 x_{k+n-1} 2^{-k} = x_n 2^{-1}$$

$$\sum_{k=1}^2 x_{k+n-2} 2^{-k} = x_{n-1} 2^{-1} + x_n 2^{-2}$$

$$\sum_{k=1}^3 x_{k+n-3} 2^{-k} = x_{n-2} 2^{-1} + x_{n-1} 2^{-2} + x_n 2^{-3}$$

⋮

$$\sum_{k=1}^n x_{k+n-n} 2^{-k} = x_1 2^{-1} + x_2 2^{-2} + x_3 2^{-3} + \cdots + x_n 2^{-n}$$

# Factorización...

## Transformada de Fourier Cuántica

Finalmente, la acción de la Transformada de Fourier Cuántica sobre un estado de la base computacional está dada por:

$$\begin{aligned} U_{FT}|x_1\rangle \otimes |x_2\rangle \otimes \cdots \otimes |x_n\rangle &= \frac{1}{\sqrt{2}}(|0_1\rangle + e^{2\pi i(x_n 2^{-1})}|1_1\rangle) \otimes \\ &\quad \frac{1}{\sqrt{2}}(|0_2\rangle + e^{2\pi i(x_{n-1} 2^{-1} + x_n 2^{-2})}|1_2\rangle) \otimes \\ &\quad \frac{1}{\sqrt{2}}(|0_3\rangle + e^{2\pi i(x_{n-2} 2^{-1} + x_{n-1} 2^{-2} + x_n 2^{-3})}|1_3\rangle) \otimes \\ &\quad \vdots \\ &\quad \frac{1}{\sqrt{2}}(|0_n\rangle + e^{2\pi i(x_1 2^{-1} + x_2 2^{-2} + x_3 2^{-3} + \cdots + x_n 2^{-n})}|1_n\rangle) \end{aligned}$$

# Factorización...

## Transformada de Fourier Cuántica

La expresión anterior sugiere una forma de implementar la Transformada de Fourier Cuántica. Para esto definimos primero el siguiente operador o compuerta cuántica:

$$R_k = |0\rangle\langle 0| + e^{2\pi i 2^{-k}} |1\rangle\langle 1|$$

Éste agrega una fase al estado  $|1\rangle$  y deja el estado  $|0\rangle$  inalterado. Notemos que la acción de  $R_k$  depende del valor del parámetro  $k$ . A modo de ejemplo tenemos:

$$R_k \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i 2^{-k}} |1\rangle)$$

# Factorización...

## Transformada de Fourier Cuántica

Otra compuerta cuántica útil es la Transformada de Hadamard, definida como:

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

Con ella podemos obtener:

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \quad \text{y} \quad H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$

o equivalentemente

$$H|x\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi ix2^{-1}}|1\rangle)$$

# Factorización...

## Transformada de Fourier Cuántica

Finalmente introducimos la Rotación Condicional (o transformación de fase condicional) de finita por:

$$C_{k,ij} = \mathbb{I}_i |0\rangle_j \langle 0| + R_{k,i} |1\rangle_j \langle 1|$$

Esta transformación aplica sobre el qubit  $i$  la compuerta cuántica  $R_k$  si el qubit  $j$  se encuentra en el estado  $|1\rangle$ . En caso contrario, si el qubit  $j$  se encuentra en el estado  $|0\rangle$ , el estado del qubit  $i$  permanece inalterado.

# Factorización...

## Transformada de Fourier Cuántica

Veamos ahora la acción de la secuencia  $C_{2,12}H_1|x_1\rangle|x_2\rangle$  en detalle.

Primero tenemos que:

$$H_1|x_1\rangle|x_2\rangle = \frac{1}{\sqrt{2}}(|0_1\rangle + e^{2\pi i x_1 2^{-1}}|1_1\rangle)|x_2\rangle$$

Luego obtenemos:

$$C_{2,12}H_1|x_1\rangle|x_2\rangle = \frac{1}{\sqrt{2}}(|0_1\rangle + e^{2\pi i (x_1 2^{-1} + x_2 2^{-2})}|1_1\rangle)|x_2\rangle$$

# Factorización...

## Transformada de Fourier Cuántica

No es difícil imaginar que para el caso de  $n$  qubits la secuencia

$$C_{n,1n} \dots C_{3,13}C_{2,12}H_1$$

transformará el primer qubit del estado

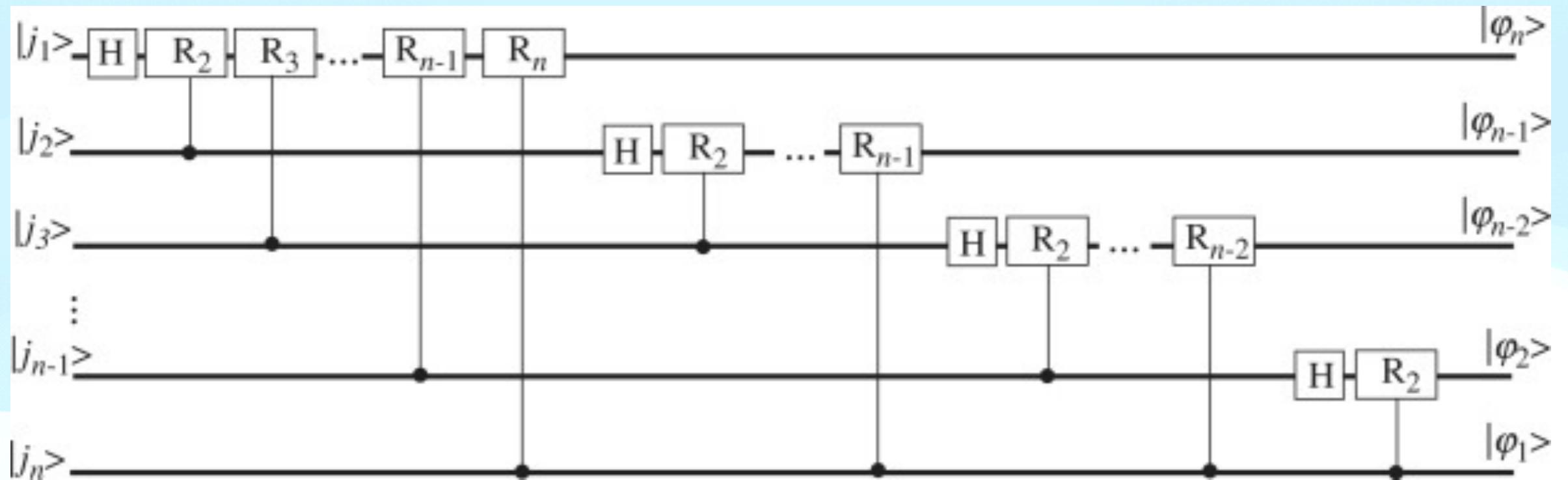
$$|x_1\rangle|x_2\rangle\dots|x_n\rangle$$

en el estado

$$\frac{1}{\sqrt{2}}(|0_1\rangle + e^{2\pi i(x_12^{-1}+x_22^{-2}+\dots+x_n2^{-n})}|1_1\rangle)|x_2\rangle\dots|x_n\rangle$$

# Factorización...

## Transformada de Fourier Cuántica



Circuito cuántico para implementar la Transformada de Fourier Cuántica.

Nótese que los estados de salida tienen el orden invertido, de modo que es necesario aplicar una secuencia de compuertas Swap después del circuito.

# Factorización...

## Transformada de Fourier Cuántica

La Transformada de Fourier Cuántica sobre  $n$  requiere  $n$  compuertas de Hadamard y  $n(n - 1)/2$  compuertas condicionales. Luego, requiere un total de  $n(n + 1)/2$  compuertas. Si consideramos las  $n/2$  compuertas Swap requeridas, donde cada una se implementa con 3 compuertas CNOT, para ordenar los estados de salida, entonces el total de compuertas es  $n^2/2 + 2n$ .

En un computador clásico la Transformada de Fourier requiere del orden de  $2^{2n}$  compuertas. La Transformada de Fourier Rápida requiere del orden de  $2^n \log(2^n)$  compuertas.

# Factorización...

## Estimación de Fase

El objetivo es encontrar el valor propio de una transformación unitaria asociado a un estado propio previamente conocido. Esto es:

para  $U$  y  $|u\rangle$  dados determinar el valor de  $0 \leq \phi \leq 1$  tal que

$$U|u\rangle = e^{2\pi i\phi}|u\rangle$$

Consideraremos dos registros cuánticos: (i) un primer registro formado por  $t$  qubits que almacena la representación binaria de la fase  $\phi$ , donde  $t$  determina la precisión con la cual la fase es representada, y (ii) un segundo registro que codifica el estado propio  $|u\rangle$ .

# Factorización...

## Estimación de Fase

El primer paso del algoritmo es estimación de fase consiste en inicializar el primer registro cuántico en el estado

$$|0_1\rangle|0_2\rangle\dots|0_t\rangle$$

y a continuación aplicar la transformación de Hadamard sobre cada uno de los qubits del registro

$$H_1|0_1\rangle H_2|0_2\rangle \dots H_t|0_t\rangle$$

obteniendo el estado

$$\frac{1}{\sqrt{2}}(|0_1\rangle + |1_1\rangle) \frac{1}{\sqrt{2}}(|0_2\rangle + |1_2\rangle) \dots \frac{1}{\sqrt{2}}(|0_t\rangle + |1_t\rangle)$$

# Factorización...

## Estimación de Fase

A continuación definimos la transformación condicional  $U_{p,i}$  por medio de

$$U_{p,i} = |0\rangle_i\langle 0| \mathbb{I} + |1\rangle_i\langle 1| U^p$$

Si el qubit  $i$  del primer registro se encuentra en el estado  $|0\rangle_i$  entonces el estado del segundo registro permanece inalterado. Si el qubit  $i$  del primer registro se encuentra en el estado  $|1\rangle_i$  entonces el estado del segundo registro es modificado por la acción de la transformación  $U^p$ . Luego tenemos

$$U_{P,i} \frac{1}{\sqrt{2}}(|0_i\rangle + |1_i\rangle) |u\rangle = \frac{1}{\sqrt{2}}(|0_i\rangle + e^{2\pi i \phi p} |1_i\rangle) |u\rangle$$

# Factorización...

## Estimación de Fase

El segundo paso es la aplicación de la secuencia de compuertas condicionales

$$U_{2^0,n} U_{2^1,n-1} U_{2^2,n-2} \dots U_{2^{t-2},2} U_{2^{t-1},1}$$

sobre el estado

$$\frac{1}{\sqrt{2}}(|0_1\rangle + |1_1\rangle) \frac{1}{\sqrt{2}}(|0_2\rangle + |1_2\rangle) \dots \frac{1}{\sqrt{2}}(|0_t\rangle + |1_t\rangle)$$

obteniendo

$$\frac{1}{\sqrt{2}}(|0_1\rangle + e^{2\pi i \phi 2^{t-1}}|1_1\rangle) \frac{1}{\sqrt{2}}(|0_2\rangle + e^{2\pi i \phi 2^{t-2}}|1_2\rangle) \dots$$

$$\frac{1}{\sqrt{2}}(|0_{t-2}\rangle + e^{2\pi i \phi 2^2}|1_{t-2}\rangle) \frac{1}{\sqrt{2}}(|0_{t-1}\rangle + e^{2\pi i \phi 2^1}|1_{t-1}\rangle) \frac{1}{\sqrt{2}}(|0_t\rangle + e^{2\pi i \phi 2^0}|1_t\rangle)$$

# Factorización...

## Estimación de Fase

El estado anterior puede ser escrito en la forma

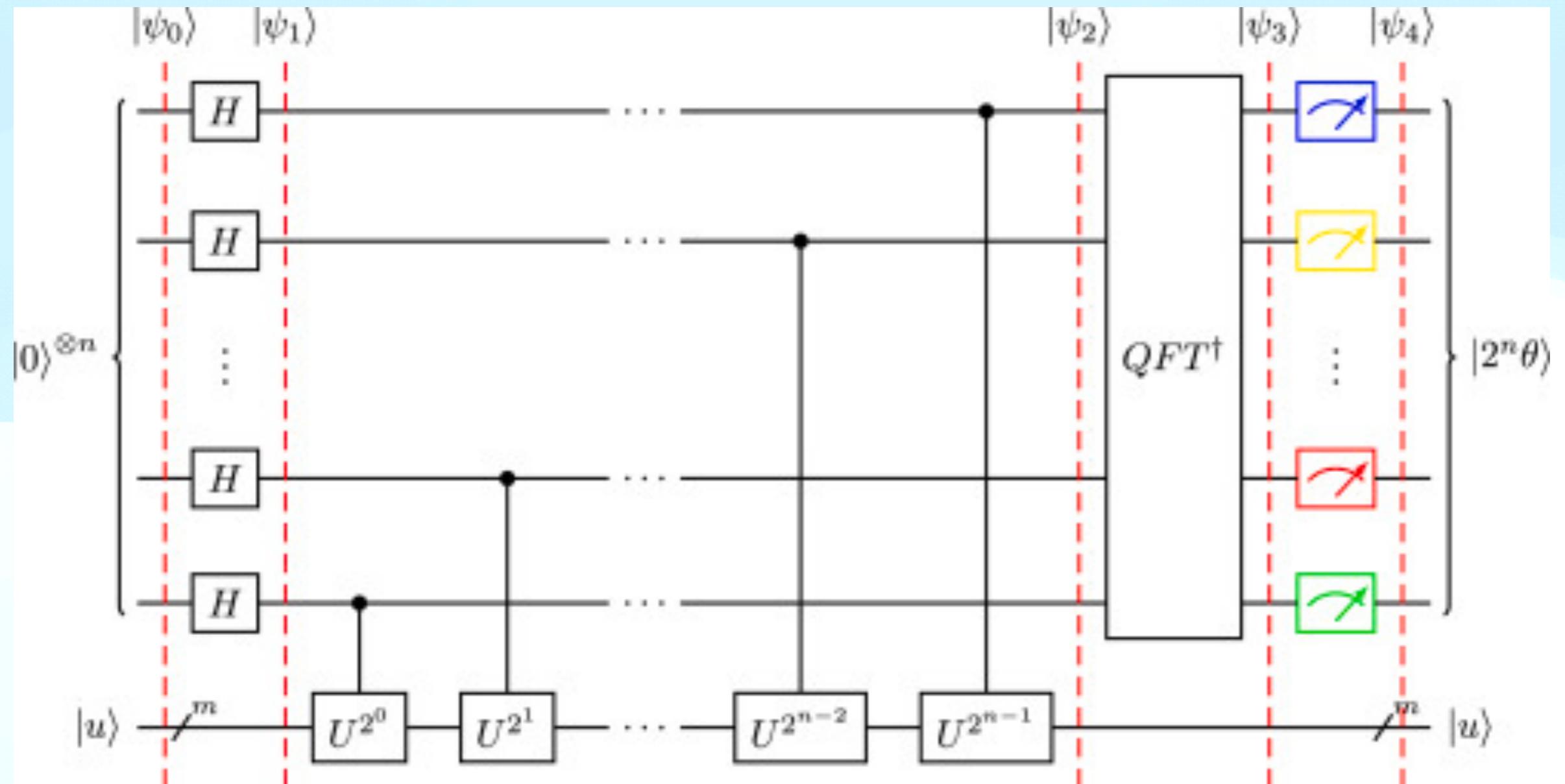
$$\left[ \frac{1}{\sqrt{2^t}} \sum_{k=0}^{2^t-1} e^{2\pi i \phi k} |k\rangle \right] |u\rangle$$

Aplicando la Transformada de Fourier Inversa a este estado obtenemos (obviando el segundo registro)

$$\frac{1}{2^t} \sum_{m=0}^{2^t-1} \left( \sum_{k=0}^{2^t-1} e^{2\pi i \phi (2^t \phi - m) k} \right) |m\rangle$$

# Factorización...

## Estimación de Fase



Circuito cuántico para Estimación de Fase

# Factorización...

## Estimación de Fase

$$\frac{1}{2^t} \sum_{m=0}^{2^t-1} \left( \sum_{k=0}^{2^t-1} e^{\frac{2\pi i}{2^t} (2^t\phi - m)k} \right) |m\rangle$$

La variable desconocida  $\phi$  es continua. Sin embargo, los estados  $|m\rangle$  sólo pueden codificar variables discretas. Para estudiar esta situación hacemos la siguiente substitución:

$$2^t\phi = a + 2^t\delta$$

Donde  $a$  es el entero mas cercano a  $2^t\phi$  y  $0 \leq |2^t\delta| \leq 1/2$ . La variable  $\delta$  cuantifica cuanto se aleja  $2^t\phi$  de un entero.

# Factorización...

## Estimación de Fase

$$\frac{1}{2^t} \sum_{m=0}^{2^t-1} \left( \sum_{k=0}^{2^t-1} e^{2\pi i(a-m)k + 2\pi i\delta k} \right) |m\rangle$$

(i) El caso más simple es  $\delta = 0$ , donde tenemos que

$$\begin{aligned} \frac{1}{2^t} \sum_{m=0}^{2^t-1} \left( \sum_{k=0}^{2^t-1} e^{2\pi i(a-m)k + 2\pi i\delta k} \right) |m\rangle &= \frac{1}{2^t} \sum_{m=0}^{2^t-1} \left( \sum_{k=0}^{2^t-1} e^{2\pi i(a-m)k} \right) |m\rangle \\ &= \frac{1}{2^t} \sum_{m=0}^{2^t-1} 2^t \delta_{a,m} |m\rangle \\ &= |a\rangle \end{aligned}$$

# Factorización...

## Estimación de Fase

Vemos entonces que cuando la fase desconocida  $2^t\phi$  es exactamente un número entero, el algoritmo de estimación de fase nos proporciona un estado de la base computacional del primer registro que codifica el valor de  $2^t\phi$ . Luego, podemos medir cada uno de los qubits del primer registro y determinar la fase desconocida.

Si  $\delta \neq 0$  entonces la probabilidad de proyectar sobre el estado  $|a\rangle$  sera

$$P(|a\rangle) = \frac{1}{2^{2t}} \left| \frac{\sin(\pi 2^t \delta)}{\sin(\pi \delta)} \right|^2 \geq \frac{4}{\pi^2} \approx 0.4052$$

Si queremos aproximar  $\phi$  con precisión de  $n$  bits con probabilidad de éxito de  $1 - \epsilon$ , entonces

$$t = n + \lceil \log(2 + \frac{1}{2\epsilon}) \rceil$$

# Factorización...

## Búsqueda de período

Consideramos una función  $f: \{0, \dots, q-1\} \rightarrow \{0, \dots, q-1\}$  tal que  $q = 2^l$ .

Esta función es periódica, esto es, si se cumple que

$$f(a) = f(a + r) \quad r \neq 0$$

$$f(a) \neq f(a + s) \quad \forall s < r$$

El objetivo es encontrar el valor del período  $r$ .

# Factorización...

## Búsqueda de período

Para resolver este problema usamos dos registros cuánticos, ambos de  $l$  qubits, previamente inicializados en el estado

$$\prod_{k=0}^{l-1} |0_k\rangle \prod_{m=0}^{l-1} |0_m\rangle$$

Luego, aplicamos la transformada de Hadamard sobre cada uno de los qubits del primer registro obteniendo

$$\prod_{k=0}^{l-1} H_k |0_k\rangle \prod_{m=0}^{l-1} |0_m\rangle = \prod_{k=0}^{l-1} \frac{1}{\sqrt{2}} (|0_k\rangle + |1_k\rangle) \prod_{m=0}^{l-1} |0_m\rangle$$

# Factorización...

## Búsqueda de período

El estado de ambos registros cuánticos se puede escribir como

$$\prod_{k=0}^{l-1} \frac{1}{\sqrt{2}} (|0_k\rangle + |1_k\rangle) \prod_{m=0}^{l-1} |0_m\rangle = \frac{1}{\sqrt{2^l}} \sum_{k=0}^{q-1} |k\rangle \prod_{m=0}^{l-1} |0_m\rangle$$

De modo que el primer registro cuántico se encuentra ahora en una superposición de todos los enteros entre 0 y  $q - 1$ .

Ahora definimos una transformación unitaria  $U$  que implementa la función  $f$ , esto es

$$U|k\rangle \prod_{m=0}^{l-1} |0_m\rangle = |k\rangle |f(k)\rangle$$

donde ahora el segundo registro cuántico codifica los valores de la función  $f$ .

# Factorización...

## Búsqueda de período

Ahora aplicamos la transformación  $U$  sobre el estado de ambos registros cuánticos y obtenemos

$$U \frac{1}{\sqrt{q}} \sum_{k=0}^{q-1} |k\rangle \prod_{m=0}^{l-1} |0_m\rangle = \frac{1}{\sqrt{q}} \sum_{k=0}^{q-1} |k\rangle |f(k)\rangle$$

En este estado el primer registro tiene un entero  $k$  y el segundo registro el valor que le asigna la función  $f$  a dicho entero, es decir,  $f(k)$ .

Si medimos el segundo registro cuántico en la base computacional encontraremos un entero, por ejemplo

$$|f(k) = z\rangle$$

# Factorización...

## Búsqueda de período

Después de la medición, ambos registros cuánticos estarán descritos por el estado

$$\left( \sqrt{\frac{r}{q}} \sum_{a|f(a)=z} |a\rangle \right) |f(a) = z\rangle$$

Esto se debe a que, dado que la función es periódica, existen varios enteros  $a \in \{0, \dots, q = 2^l - 1\}$  tales que  $f(a) = z$ .

# Factorización...

## Búsqueda de período

Ahora definimos el siguiente entero

$$a_0 = \min\{a | f(a) = z\}$$

y escribimos el estado del primer registro cuántico como

$$\sqrt{\frac{r}{q}} \sum_{t=0}^{q/r-1} |a_0 + tr\rangle$$

donde hemos asumido que  $r$  divide a  $q$ .

# Factorización...

## Búsqueda de período

Podemos entonces aplicar la Transformada de Fourier Cuántica sobre el estado del primer registro

$$\begin{aligned} U_{FT}^{-1} \sqrt{\frac{r}{q}} \sum_{t=0}^{q/r-1} |a_0 + tr\rangle &= \sqrt{\frac{r}{q}} \sum_{t=0}^{q/r-1} U_{FT}^{-1} |a_0 + tr\rangle \\ &= \sqrt{\frac{r}{q}} \sum_{t=0}^{q/r-1} \frac{1}{\sqrt{q}} \sum_{k=0}^{q-1} \omega^{-k(a_0+tr)} |k\rangle \\ &= \sqrt{\frac{r}{q^2}} \sum_{k=0}^{q-1} e^{-2\pi i \frac{ka_0}{q}} \left( \sum_{t=0}^{q/r-1} e^{-2\pi i t \frac{kr}{q}} \right) |k\rangle \end{aligned}$$

# Factorización...

## Búsqueda de período

Supongamos ahora que  $kr/q$  es igual a un entero  $m$ . En este caso, tenemos que para un  $k$  particular se cumple que:

$$\sum_{t=0}^{q/r-1} e^{-2\pi it \frac{kr}{q}} = \sum_{t=0}^{q/r-1} e^{-2\pi itm} = \frac{q}{r}$$

y la probabilidad de medir un estado del primer registro con  $k = mq/r$  es

$$P(k = m\frac{q}{r}) = \frac{1}{r}$$

Luego, la probabilidad de medir algún estado con  $k = mq/r$  es

$$\sum_{k=m\frac{q}{r}} P(k = m\frac{q}{r}) = \frac{1}{r}r = 1$$

# Factorización...

## Búsqueda de período via estimación de fase

Supongamos ahora que  $kr/q$  es igual a un entero  $m$ . En este caso, tenemos que para un  $k$  particular se cumple que:

$$\sum_{t=0}^{q/r-1} e^{-2\pi it \frac{kr}{q}} = \sum_{t=0}^{q/r-1} e^{-2\pi itm} = \frac{q}{r}$$

y la probabilidad de medir un estado del primer registro con  $k = mq/r$  es

$$P(k = m\frac{q}{r}) = \frac{1}{r}$$

Luego, la probabilidad de medir algún estado con  $k = mq/r$  es

$$\sum_{k=m\frac{q}{r}} P(k = m\frac{q}{r}) = \frac{1}{r}r = 1$$

# Factorización...

## Algoritmo de Shor

En 1994, **Peter Shor**, un matemático aplicado del Instituto Tecnológico de Massachusetts (MIT), mostró como un computador cuántico podría encontrar los factores primos de grandes números enteros en un tiempo exponencialmente mucho menor que por medio de un computador clásico.

La aplicación del algoritmo de Shor fue obvia e inmediata: los números primos son usados como llaves que proporcionan la seguridad en la mayoría de la información encriptada transmitida sobre la internet. Luego, el algoritmo de Shor tiene el potencial para decriptar comunicaciones supuestamente seguras.

Para un entero de  $n$  bits el algoritmo de Shor requiere  $n^2$  compuertas. Actualmente se emplean 2048 bits para encriptar las comunicaciones, de modo que el algoritmo de Shor usaría cerca de 4 millones de compuertas. Los prototipo de computadores cuánticos actuales tiene sólo cientos de qubits y están muy lejos de ejecutar millones de compuertas.

# Factorización...

## Algoritmo de Shor

1. ¿Es  $N$  par?, si es el caso entregar como resultado 2 y parar.
2. ¿Es  $N = c^l$  para enteros  $c$  y  $l$ ?, si es el caso entregar  $c$  como resultado y parar.
  - Para los pasos 1 y 2 existen algoritmos clásicos **eficientes**.
3. Si  $N$  no es impar ni potencia de un primo, entonces escoger aleatoriamente un entero  $x$  tal que  $1 \leq x \leq N$  y calcular  $s = gcd(x, N)$ , esto es, el divisor común exacto (sin residuo) mas grande entre  $x$  y  $N$ . Si  $s \neq 1$  entonces entregar como resultado  $s$ , pues es un factor de  $N$ , y parar.
  - Para el paso 3 se puede emplear el algoritmo de división de Euclides, el cual es **eficiente**.

# Factorización...

## Algoritmo de Shor

4. Dado que  $s = 1$  los enteros  $x$  y  $N$  son co-primos, es decir, no tienen ningún divisor en común, excepto 1. Encontrar el orden  $r$  de la función  $x \ mod(N)$ .

El orden  $r$  es el entero más pequeño tal que se cumple la ecuación  $x^r = 1 \ mod(N)$ .

- La búsqueda del orden  $r$  se realiza en un computador cuántico por medio del algoritmo de estimación fase.

# Factorización...

## Algoritmo de Shor

5. Si  $r$  es impar, volver al paso 3 y escoger aleatoriamente un entero  $x$  distinto. Si  $r$  es par entonces usar post-procesamiento clásico para extraer un factor de  $N$  usando  $r$  y parar.

$$x^r = 1 \pmod{N} \rightarrow (x^{\frac{r}{2}} + 1)(x^{\frac{r}{2}} - 1) = 0 \pmod{N}$$

- A.  $x^{r/2} = 1 \pmod{N}$  no es posible
- B.  $x^{r/2} = -1 \pmod{N}$  el algoritmo falla
- C.  $(x^{r/2} + 1)(x^{r/2} - 1) = N \pmod{N}$ , de modo que  $(x^{r/2} \pm 1)$  son factores primos de  $N$ .
- D.  $(x^{r/2} + 1)(x^{r/2} - 1) = kN \pmod{N}$   $k \geq 2$ , en cuyo caso  $\gcd((x^{r/2} \pm 1), N)$  es un factor no trivial de  $N$ .

# Factorización...

## Algoritmo de Shor

1. En este caso  $N = 15$  no es par.
2. En este caso  $N = 15$  no es potencia de un primo.
3. Escogemos  $x = 4$  y calculamos  $s = \gcd(4, 15)$ . Obtenemos  $s = 1$ .
4. Ahora necesitamos obtener el orden  $r$ , esto es encontrar la solución  $r$  de  $4^r \equiv 1 \pmod{15}$ . Claramente,  $r = 2$  puesto que  $4^2 = 16 = 15 + 1 = 1$ .
5. Tenemos ahora que  $r/2 = 2/2 = 1$ , de modo que los factores son  $(4 \pm 1)$ , es decir 5 y 3, y son tales que  $3 \cdot 5 = 15$ .

# Factorización...

## Algoritmo de Shor

1. En este caso  $N = 15$  no es par.
2. En este caso  $N = 15$  no es potencia de un primo.
3. Escogemos  $x = 2$  y calculamos  $s = \gcd(2, 15)$ . Obtenemos  $s = 1$ .
4. Ahora necesitamos obtener el orden  $r$ , esto es encontrar la solución  $r$  de  $2^r \equiv 1 \pmod{15}$ . Claramente,  $r = 4$  puesto que  $2^4 = 16 = 15 + 1 = 1$ .
5. Tenemos ahora que  $r/2 = 4/2 = 2$ , de modo que los factores son  $(2^2 \pm 1)$ , es decir 5 y 3, y son tales que  $3 \cdot 5 = 15$ .

# Factorización...

## Algoritmo de Shor

1. En este caso  $N = 15$  no es par.
2. En este caso  $N = 15$  no es potencia de un primo.
3. Escogemos  $x = 7$  y calculamos  $s = \gcd(7, 15)$ . Obtenemos  $s = 1$ .
4. Ahora necesitamos obtener el orden  $r$ , esto es encontrar la solución  $r$  de  $7^r \equiv 1 \pmod{15}$ . Claramente,  $r = 4$  puesto que  $7^4 = 2401 = 160 \cdot 15 + 1 = 1$ .
5. Tenemos ahora que  $r/2 = 4/2 = 2$ , de modo que los factores son  $(7^2 \pm 1)$ , es decir 50 y 48, y son tales que  $50 \cdot 48 = 2400 = 160 \cdot 15$ . Tenemos  $\gcd(50, 15) = 5$  y  $\gcd(48, 15) = 3$ .

# Factorización...

## Algoritmo de Shor

Definimos ahora la función  $f(y) = x \cdot y \ mod(N)$  y su implementación por medio de la transformación  $U$  definida como:

$$U|y\rangle = |y \cdot x\rangle \ mod(N) \quad \forall 0 \leq y \leq N - 1$$

$$U|y\rangle = |y\rangle \quad \forall N \leq y \leq 2^n - 1$$

Cuando  $x$  y  $N$  son co-primos la transformación  $U$  es unitaria. Es más, es una permutación.

# Factorización...

## Algoritmo de Shor

Los estados propios de la transformación unitaria  $U$  están dados por la expresión

$$|u_s\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} e^{-\frac{2\pi i}{r} sk} |x^k\rangle \bmod(N)$$

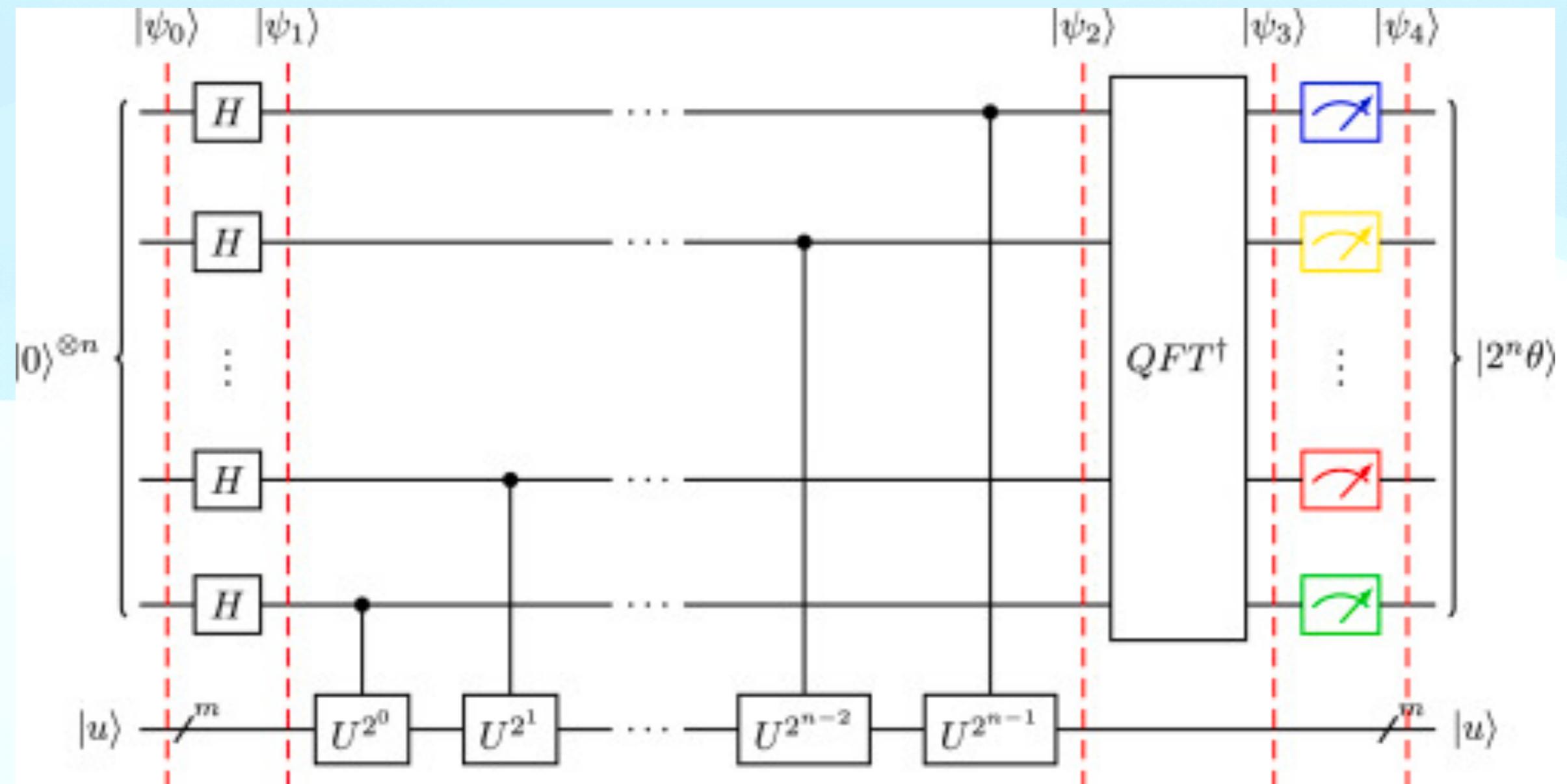
donde el valor propio es

$$U|u_s\rangle = e^{\frac{2\pi i}{r} s}|u_s\rangle$$

y claramente es una función del orden  $r$  que buscamos.

# Factorización...

## Algoritmo de Shor



# Factorización...

## Algoritmo de Shor

Para aplicar el algoritmo de estimación de fase necesitamos:

1. Poner el segundo registro cuántico en el estado propio respectivo. Cualquiera de ellos sirve puesto que todos ellos tienen valores propios de dependen de del rango  $r$ . Sin embargo, el estado propio también depende de  $r$ , el cual es desconocido y por lo tanto no podemos preparar. La solución es preparar el segundo registro en el estado

$$|1\rangle = \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |u_s\rangle$$

2. Una implementación eficiente de las compuertas condicionales  $U^{2^j}$ . Es posible hacerlo con  $O(n^3)$  compuertas.

# Factorización...

## Algoritmo de Shor

Entonces, el algoritmo de estimación de fase efectuará la siguiente transformación

$$|0_0\rangle \otimes \cdots \otimes |0_{n-1}\rangle |1\rangle = \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |s/r\rangle |u_s\rangle$$

Vemos que obtenemos en el primer registro una superposición de estados que codician la fase  $s/r$  para distintos valores de  $r$ . Al medir este registro obtendremos uno de los valores  $s/r$ , desde el cual podemos obtener el valor del orden  $r$  por medio del algoritmo clásico de fracciones continuadas con una probabilidad alta.

# Factorización...

## Algoritmo de Shor

Una vez que tenemos el valor del orden  $r$  podemos proceder a calcular los factores de  $N$ . Pero, como vimos anteriormente, tenemos la siguiente situación:

1. Existen dos casos en los cuales fallamos. La probabilidad de que esto no ocurra es al menos de  $1/2$ .
2. Es posible que el algoritmo empleado para extraer el orden  $r$  desde la fase  $s/r$  falle. Sin embargo, la probabilidad de que esto no ocurra es del orden de  $1 - \epsilon$ .

Luego, la probabilidad de éxito del algoritmo de Shor es del orden de  $O(1)$  y no crece con  $n$ .

# Factorización...

## Algoritmo de Shor

Una vez que tenemos el valor del orden  $r$  podemos proceder a calcular los factores de  $N$ . Pero, como vimos anteriormente, tenemos la siguiente situación:

1. Existen dos casos en los cuales fallamos. La probabilidad de que esto no ocurra es al menos de  $1/2$ .
2. Es posible que el algoritmo empleado para extraer el orden  $r$  desde la fase  $s/r$  falle. Sin embargo, la probabilidad de que esto no ocurra es del orden de  $1 - \epsilon$ .

Luego, la probabilidad de éxito del algoritmo de Shor es del orden de  $O(1)$  y no crece con  $n$ .

# Factorización...

## Algoritmo de Shor

1. El algoritmo de Shor usa dos algoritmos clásicos: uno para probar si  $N$  es potencia de un primo y el algoritmo de Euclides. Ambos requieren un número polinomial de compuertas.
2. El algoritmo de estimación de fase requiere del orden de  $O(n^3)$  compuertas mientras la transformada de Fourier cuántica del orden de  $O(n^2)$ .
3. El número de iteraciones del algoritmo de Shor requerido para tener una probabilidad alta de éxito no crece con  $n$ .

Luego, el algoritmo de Shor puede factorizar en un número de operaciones que es polinomial en el número de bits requerido para codificar el número a factorar. En contraste, el mejor algoritmo conocido para primos grandes requiere un número de operaciones dado por  $e^{O(n^{1/3} \log^{2/3} n)}$ .

# Factorización...

## Algoritmo de Shor

### An Efficient Quantum Factoring Algorithm

Oded Regev\*

#### Abstract

We show that  $n$ -bit integers can be factorized by independently running a quantum circuit with  $\tilde{O}(n^{3/2})$  gates for  $\sqrt{n} + 4$  times, and then using polynomial-time classical post-processing. The correctness of the algorithm relies on a number-theoretic heuristic assumption reminiscent of those used in subexponential classical factorization algorithms. It is currently not clear if the algorithm can lead to improved physical implementations in practice.

arXiv:2308.06572v2 [quant-ph] 17 Aug 2023

