Università degli Studi di Trieste

DIPARTIMENTO DI FISICA
Corso di Laurea in Fisica

Tesi di laurea

L'algoritmo di fattorizzazione di Shor e la crittografia quantistica

Candidato:
Giovanni Varutti

Relatore:
Prof. Fabio Benatti

Anno Accademico 2021-2022

Sommario

Nel maggio del 1981, in una conferenza chiamata *'Physics of Computation'*, organizzata dal MIT di Boston e da IBM, si riunirono alcuni dei fisici e degli informatici più affermati dell'epoca. Gli interventi che presero parte alla conferenza furono pubblicati un anno dopo sulla rivista *International Journal of Theoretical Physics*. Uno dei più significativi fu proposto da Richard Feynman, che teorizzò l'esistenza di un nuovo tipo di macchina computazionale, il computer quantistico, il cui scopo principale fosse quello di simulare sistemi fisici quantistici, compito che risultava inadatto per i tradizionali computer classici. Quest'ultimi infatti vanno incontro ad un rallentamento esponenziale, cosa che impedisce di eseguire una simulazione efficiente di un sistema quantistico. La computazione, infatti, viene vista come un processo fisico, e al fine di ottenere una simulazione efficiente è necessario che le risorse utilizzate dal computer siano proporzionali alle dimensioni del sistema fisico da simulare. Il discorso tenuto da Feynman fu poi pubblicato nel suo articolo *'Simulating physics with computers'* [16], che divenne così uno dei punti di riferimento della computazione quantistica, e viene da molti considerato simbolicamente come la nascita della disciplina. Negli anni successivi le idee di Feynman presero piede e in molti si posero il problema se i computer quantistici potessero avere un vantaggio computazionale rispetto ai computer classici tradizionali, anche nel risolvere problemi non esplicitamente legati alla simulazione di sistemi quantistici. Un punto di svolta per la computazione quantistica avvenne nel 1994, quando Peter Shor pubblicò un articolo [5] in cui viene illustrato un algoritmo quantistico capace di fattorizzare un numero intero in un tempo polinomiale, ossia in maniera esponenzialmente più veloce rispetto ad ogni algoritmo classico conosciuto. Il seguente lavoro di tesi ha lo scopo di presentare nel dettaglio l'algoritmo di Shor, ed è ispirato al problema posto da Feynman, che dopo più di quarant'anni risulta ancora estremamente attuale: quanto è possibile sfruttare la natura quantistica della materia per innovare o rivoluzionare il modo in cui vengono elaborati i dati e risolti i problemi numericamente. Nel primo capitolo vengono introdotti i principali elementi della computazione quantistica, dal qubit alla trasformata di Fourier quantistica. Il secondo capitolo è dedicato all'analisi dettagliata dei vari strumenti matematici che costituiscono l'algoritmo di Shor, con particolare riferimento alla *Quantum Phase Estimation* e all'*Order Finding*. Nell'ultimo capitolo si affrontano la crittografia RSA e la crittografia quantistica, evidenziando le implicazioni pratiche dell'algoritmo di Shor nell'ambito della sicurezza informatica.

Indice

Capitolo 1

Introduzione al Quantum Computing

Al fine di illustrare efficacemente sia il formalismo matematico su cui si basa l'algoritmo di Shor, sia i concetti che ne determinano l'importanza e le varie implicazioni, è necessario fare un'introduzione teorica sui principali argomenti del Quantum Computing. Questa prima parte della trattazione fa in gran parte riferimento al libro di Nielsen-Chuang [1], e quello di Nakahara-Ohmi [3].

1.1 Bit Quantistici

L'elemento fondamentale della computazione classica è il bit, ossia un sistema numerico binario che può assumere solamente due valori, 0 e 1. Tutta l'informazione classica viene tradotta e trasmessa usando bit. La computazione quantistica generalizza questo sistema, in quanto l'elemento base di quest'ultima è un generico vettore bi-dimensionale normalizzato appartenete allo spazio di Hilbert $\mathcal{H} = \mathbb{C}^2$, chiamato bit quantistico o, in breve, qubit:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \in \mathcal{H},\ \text{con}\ \ |0\rangle = \begin{pmatrix}1\\0\end{pmatrix}\ \text{e}\ \ |1\rangle = \begin{pmatrix}0\\1\end{pmatrix}, \tag{1.1.1}$$

dove le ampiezze di probabilità $\alpha, \beta \in \mathbb{C}$ con $|\alpha|^2 + |\beta|^2 = 1$, e $\{|0\rangle, |1\rangle\}$ è la base ortonormale standard di $\mathbb{C}^2$, denominata base computazionale. É chiaro dunque che, grazie alla sovrapposizione quantistica degli stati, il qubit non è limitato come il bit classico ad un semplice sistema binario. Per realizzare fisicamente un qubit possono essere sfruttate proprietà quantistiche come la polarizzazione dei fotoni: in questo caso lo stato $|0\rangle$ rappresenta la polarizzazione orizzontale, mentre $|1\rangle$ quella verticale. Usando per esempio dei filtri polarizzatori di varie orientazioni si possono mettere in sovrapposizione questi stati e creare una combinazione lineare (polarizzazione circolare) come

nell'equazione (1.1.1). Può essere utile, per vari scopi, dare anche un'interpretazione geometrica del qubit, usando la rappresentazione di Bloch. Siccome α e β sono numeri complessi e vale l'identità $|\alpha|^2 + |\beta|^2 = 1$, si può scrivere lo stato dell'equazione (1.1.1) in funzione di due parametri reali:

$$|\psi(\theta, \varphi)\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\varphi}\sin\frac{\theta}{2}|1\rangle \tag{1.1.2}$$

al netto di una fase moltiplicativa, trascurata in quanto non produce effetti osservabili. Nelle coordinate sferiche, i numeri reali θ e φ rappresentano quindi un punto su una sfera unitaria tridimensionale, chiamata sfera di Bloch; tale punto corrisponde allo stato generico $|\psi\rangle$ del qubit considerato.

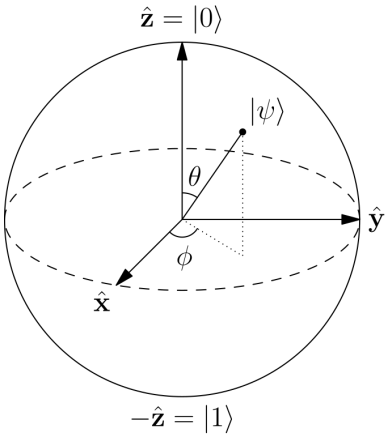


Figura 1.1.1: Sfera di Bloch

Nel caso di un sistema di più qubit bisogna considerare il prodotto tensore di operatori e spazi di Hilbert. In generale, per un sistema di n qubit rappresentati dai vettori nell'equazione (1.1.1), lo spazio di Hilbert associato è $\mathcal{H} = \mathbb{C}^2 \otimes \ldots \otimes \mathbb{C}^2 = \mathbb{C}^{2^n}$, e la sua base computazionale è l'insieme di 2^n vettori ortogonali:

$$\{|x_1\rangle \otimes \ldots \otimes |x_n\rangle \equiv |x_1 \ldots x_n\rangle\}_{x_1,\ldots,x_n=0,1} \quad .$$

Lo stato del sistema si può quindi scrivere come una generica combinazione lineare di tali vettori. Inoltre, se lo stato del sistema si può scrivere in maniera fattorizzata come

$$|\psi\rangle = |\psi_1\rangle \otimes \ldots \otimes |\psi_n\rangle = |\psi_1 \ldots \psi_n\rangle \quad ,$$

allora è detto separabile. In caso contrario viene chiamato stato *entangled*. Si può anche passare dal sistema binario a quello decimale: sia dunque $N = 2^n$ e $x = x_1 2^{n-1} + x_2 2^{n-2} + \ldots + x_n 2^0$ con $x_1, \cdots, x_n = 0, 1$. In questo caso

gli elemnti della base computazinale si possono scrivere nella forma $|x\rangle$ con $x = 0, 1, \ldots, N-1$ e uno stato generico del sistema come:

$$|\psi\rangle = \sum_{x=0}^{N-1} \alpha_x |x\rangle \quad , \tag{1.1.3}$$

con associato il vettore delle $N = 2^n$ ampiezze di probabilità, o coefficienti di Fourier:

$$\vec{\alpha} = \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_{N-1} \end{pmatrix} = \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_{2^n-1} \end{pmatrix} \quad .$$

Fondamentale, nell'ambito della computazione quantistica, è anche il procedimento di misura. Matematicamente, si introduce un operatore di misura M_m su $\mathbb{C}^{\mathbb{N}}$, tale che, dopo il processo di misura effettuato sullo stato $|\psi\rangle$, si ottenga lo stato finale $|\psi_m\rangle$ con una probabilità $P(m)$ data da:

$$|\psi_m\rangle = \frac{M_m |\psi\rangle}{\sqrt{P(m)}} \quad , \quad P(m) = \langle\psi| M_m^\dagger M_m |\psi\rangle \, . \tag{1.1.4}$$

Nel caso esemplificativo di due qubit, la base computazionale standard di $\mathbb{C}^2 \otimes \mathbb{C}^2 = \mathbb{C}^4$ è $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$, quindi il generico stato è dato dalla seguente combinazione lineare:

$$|\psi\rangle = \alpha_{00} |00\rangle + \alpha_{01} |01\rangle + \alpha_{10} |10\rangle + \alpha_{11} |11\rangle \, . \tag{1.1.5}$$

Eseguendo un processo di misura su entrambi i qubit, lo stato del sistema collasserebbe in uno dei vettori $M_{ij} = |ij\rangle \langle ij|$, $i, j = 0, 1$, relativo alla base computazionale $|ij\rangle$. Eseguendo invece una misura $M_i = |i\rangle\langle i|$, $i = 0, 1$, relativa alla base computazionale del solo primo qubit, ed ottenendo come esito $|1\rangle$, l'operatore di misura su entrambi i qubit diventa $M_1 = |1\rangle \langle 1| \otimes \mathbb{1}$ e lo stato finale dopo la misura eseguita su $|\psi\rangle$ in (1.1.5) risulta

$$|\psi_1\rangle = \frac{\alpha_{10} |10\rangle + \alpha_{11} |11\rangle}{\sqrt{|\alpha_{10}|^2 + |\alpha_{11}|^2}} \quad .$$

Come visto, per un sistema di n qubit si possono costruire stati con 2^n ampiezze di probabilità. A differenza quindi della computazione classica, in cui è sufficiente memorizzare gli n bit binari, nella computazione quantistica è necessario invece costruire un registro quantistico per memorizzare le 2^n ampiezze di probabilità, che sono, peraltro, numeri complessi. Infatti, se qualcuno volesse costruire esattamente lo stato $|\psi\rangle$ del sistema di n qubit, dovrebbe conoscere e memorizzare tutta l'informazione in esso contenuta, ossia le 2^n ampiezze di probabilità. Tuttavia questa informazione non è accessibile in maniera diretta a chi osserva o effettua misure sullo stato $|\psi\rangle$

senza conoscere $\vec{\alpha}$, in quanto un processo di misura (su tutti i qubit) fa collassare lo stato in uno degli elementi della base computazionale, con una certa probabilità data dal modulo quadro del corrispondente coefficiente di Fourier. Bisognerebbe disporre idealmente di infinite copie dello stato $|\psi\rangle$ ed effettuare su di esso infinite misure per ricostruire $\vec{\alpha}$. Tale possibilità è impedita dal teorema di no-cloning quantistico, discusso nella sezione 1.3.

1.2 Quantum Gates

Un altro elemento fondamentale per introdurre la computazione quantistica sono i quantum gates, ossia l'equivalente quantistico delle porte logiche classiche. Siccome lo stato del singolo qubit (1.1.1), o di un sistema di più qubits (1.1.3), deve restare normalizzato, al fine di conservarne la probabilità totale, è possibile agire su di esso solamente con operatori che ne preservino la norma, ossia operatori unitari su spazi di Hilbert. I gates quantistici sono quindi rappresentati da matrici unitarie. Da questo segue inoltre che i gates quantistici sono tutti reversibili, a differenza di quelli classici, in quanto una matrice $U : \mathcal{H} \mapsto \mathcal{H}$ è unitaria se $U^{-1} = U^{\dagger}$ ed è quindi anche invertibile. I gates quantistici più semplici sono quelli che agiscono su un singolo qubit, e pertanto sono rappresentati da matrici unitarie 2×2. Quelli più utilizzati nell'ambito della computazione quantistica sono i seguenti:

Hadamard $-\boxed{H}-$ $\dfrac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}$ ‖ Pauli-X $-\boxed{X}-$ $\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}$

Pauli-Y $-\boxed{Y}-$ $\begin{bmatrix}0 & -i\\ i & 0\end{bmatrix}$ ‖ Pauli-Z $-\boxed{Z}-$ $\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}$

Phase-k $-\boxed{R_k}-$ $\begin{bmatrix}1 & 0\\ 0 & e^{2\pi i/2^k}\end{bmatrix}$

In particolare il gate dato dalla matrice di Pauli X è la generalizzazione quantistica del NOT gate classico, in quanto la sua azione sulla base computazionale, $X|0\rangle = |1\rangle, X|1\rangle = |0\rangle$, inverte i qubits di quest'ultima. Fondamentale importanza riveste anche il gate di Hadamard, in quanto è in grado di creare una sovrapposizione quantistica fra gli stati $|0\rangle, |1\rangle$ della base computazionale:

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

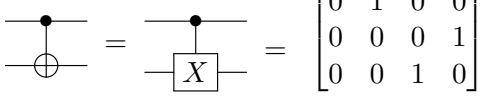
Essendo H unitaria, gli stati $\{|+\rangle, |-\rangle\}$ formano una base ortonormale di $\mathbb{C}^2$ chiamata anche base di Fourier. Il gate R_k, sempre a singolo qubit, agisce sulla base computazionale nel modo seguente:

$$R_k|j\rangle = e^{2\pi i j/2^k}|j\rangle, j = 0, 1 \implies R_k|1\rangle = e^{2\pi i/2^k}|1\rangle, \quad R_k|0\rangle = |0\rangle$$

ossia applica una fase $e^{2\pi i/2^k}$ solamente quando lo stato del qubit è $|1\rangle$. L'azione sulla base di Fourier è dunque:

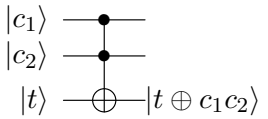
$$R_k |\pm\rangle = \frac{1}{\sqrt{2}} \left(|0\rangle \pm e^{2\pi i/2^k} |1\rangle \right).$$

Invece, per due qubit importante è il Controlled-Not gate ($CNOT$) che prende come input un qubit di controllo, $|c\rangle$, e uno di target, $|t\rangle$: se il qubit di controllo si trova nello stato $|1\rangle$ della base computazionale, allora viene applicato il NOT gate (o X gate) al target qubit, se invece il control qubit si trova nello stato $|0\rangle$ il target qubit rimane inalterato. Nella base computazionale la sua azione è data da $|c\rangle |t\rangle \to |c\rangle |t \oplus c\rangle$, dove $\oplus$ rappresenta l'addizione modulo 2. I simboli circuitali e la rappresentazione matriciale sono i seguenti:

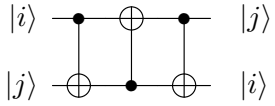


$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Il $CNOT$ gate può essere facilmente generalizzato facendo agire sul target qubit un generico gate unitario U, ottenendo il gate Controlled-U (CU). Utilizzando invece due qubits di controllo e uno di target si ottiene il gate di Toffoli, che funziona con lo stesso meccanismo sopra descritto. La sua azione sulla base computazionale è allora data da $|c_1\rangle |c_2\rangle |t\rangle \to |c_1\rangle |c_2\rangle |t \oplus c_1 c_2\rangle$ e la sua rappresentazione circuitale è:



Il gate di Toffoli riveste una certa importanza in quanto, nella computazione classica, fornisce delle versioni reversibili dei gates AND e OR, che assieme al NOT-gate formano un set di gates classici universali. Da ciò si deduce che qualsiasi circuito classico può essere simulato usando un circuito quantistico. Un set di gates universali può essere trovato anche nella computazione quantistica, e ciò permette di esprimere, o approssimare, qualsiasi circuito quantistico utilizzando tali gates. Infatti si può dimostrare [1] che qualsiasi matrice unitaria $U : \mathcal{H} \mapsto \mathcal{H}$ che agisce su uno spazio di Hilbert n-dimensionale $\mathcal{H} \equiv \mathbb{C}^n$ (e quindi qualsiasi gate quantistico su n qubits) può essere espressa esattamente usando gates a singolo qubit e il $CNOT$ gate. Può invece solo essere approssimata, con precisione arbitraria, usando i gates H, R_2, R_3 e $CNOT$, che quindi costituiscono un set di gates universali per la computazione quantistica. Un altro gate a due qubits interessante, che verrà citato nel seguito, è lo $SWAP$-gate, che scambia lo stato di due qubits sui quali è applicato.



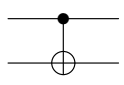
L'azione di questo circuito sullo stato iniziale è infatti:

$$\begin{aligned} |i,j\rangle &\to |i,(i\oplus j)\rangle \\ &\to |i\oplus(i\oplus j), i\oplus j\rangle = |j, i\oplus j\rangle \\ &\to |j,(i\oplus j)\oplus j\rangle = |j,i\rangle \ . \end{aligned}$$

1.3 Teorema No-Cloning

Nella computazione classica copiare un bit è un'operazione piuttosto semplice: è sufficiente disporre di un $CNOT$ gate classico con il secondo bit inizializzato a 0. Nella computazione quantistica invece, copiare esattamente lo stato in cui si trova un qubit è un'operazione non sempre permessa, coerentemente col principio di indeterminazione. Si consideri ad esempio il $CNOT$ gate quantistico con un qubit nel generico stato $|\psi\rangle$ da copiare nel secondo qubit inizializzato nello stato $|0\rangle$:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \qquad \alpha|00\rangle + \beta|11\rangle\,.$$
$$|0\rangle$$



Come si può vedere il CNOT gate quantistico non è in grado di copiare nel secondo registro lo stato generico $|\psi\rangle$: in questo caso si dovrebbe ottenere come stato finale $|\psi\rangle|\psi\rangle = |\alpha|^2|00\rangle + \alpha\beta|01\rangle + \beta\alpha|10\rangle + |\beta|^2|11\rangle$. Una dimostrazione rigorosa di questo principio può essere fatta cercando di costruire una matrice unitaria U capace di copiare lo stato $|\psi\rangle$ del primo qubit sul secondo registro $|t\rangle$: $|\psi\rangle|t\rangle \xrightarrow{U} |\psi\rangle|\psi\rangle$. Si suppone inoltre che tale relazione sia verificata almeno per due specifici stati del primo qubit $|\psi_1\rangle$ e $|\psi_2\rangle$. Si ha quindi:

$$\begin{aligned} U(|\psi_1\rangle\otimes|t\rangle) &= |\psi_1\rangle\otimes|\psi_1\rangle \\ U(|\psi_2\rangle\otimes|t\rangle) &= |\psi_2\rangle\otimes|\psi_2\rangle \end{aligned}$$

Prendendo il prodotto scalare fra i rispettivi membri delle due equazioni si ottiene, ricordando che $U^\dagger U = \mathbb{1}$ e $\|t\|^2 = \langle t|t\rangle = 1$:

$$\langle\psi_1\otimes t|\,U^\dagger U\,|\psi_2\otimes t\rangle = \langle\psi_1\otimes\psi_1|\,\psi_2\otimes\psi_2\rangle \implies \langle\psi_1|\,\psi_2\rangle = (\langle\psi_1|\,\psi_2\rangle)^2$$

$$\implies \langle\psi_1|\,\psi_2\rangle = 1 \vee \langle\psi_1|\,\psi_2\rangle = 0 \ .$$

Il primo caso è triviale in quanto equivale a $|\psi_1\rangle = |\psi_2\rangle$. Dal secondo caso si può invece dedurre che $|\psi_1\rangle \perp |\psi_2\rangle$, ossia il gate U è in grado di copiare solo stati ortogonali al primo qubit. Si può quindi costruire una matrice U in grado di copiare due stati della base computazionale o della base di Fourier, ma non in grado di copiare sia uno stato $\in \{|0\rangle, |1\rangle\}$ che uno stato $\in \{|+\rangle, |-\rangle\}$. Si supponga ora di avere due diversi stati $|\varphi_1\rangle, |\varphi_2\rangle \in \mathcal{H}$, non ortogonali, ossia $\langle\varphi_1|\varphi_2\rangle \neq 0, 1$. Si vuole cercare di distinguerli evitando

di perturbare il sistema con la misurazione. A questo scopo si prepara un qubit ancilla nello stato $|u\rangle$ e un operatore unitario U che agisca sul sistema risultante nel seguente modo:

$$|\varphi_1\rangle\,|u\rangle \to U\,|\varphi_1\rangle\,|u\rangle = |\varphi_1\rangle\,|v_1\rangle$$

$$|\varphi_2\rangle\,|u\rangle \to U\,|\varphi_2\rangle\,|u\rangle = |\varphi_2\rangle\,|v_2\rangle$$

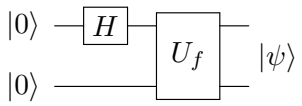
Leggendo solamente il secondo qubit, qualora $v_1 \neq v_2$, si riesce a distinguere fra i due stati $|\varphi_1\rangle\,, |\varphi_2\rangle$ senza perturbarli. Tuttavia, prendendo il prodotto scalare delle precedenti equazioni si ottiene:

$$\langle\varphi_1|\varphi_2\rangle\,\langle u|u\rangle = \langle\varphi_1|\varphi_2\rangle\,\langle v_1|v_2\rangle \implies \langle v_1|v_2\rangle = \langle u|u\rangle = 1 \implies |v_1\rangle = |v_2\rangle$$

$\implies$ qualsiasi tentativo di distinguere fra tali $|\varphi_1\rangle\,, |\varphi_2\rangle$ implica necessariamente un disturbo del sistema considerato.

1.4 Parallelismo quantistico

Il parallelismo quantistico è, come può suggerire il nome, l'abilità di un circuito quantistico di poter valutare una generica funzione $f(x)$ per diversi valori di x contemporaneamente. Nel seguito se ne vedranno quindi le potenzialità e i limiti. Si considera dunque una funzione $f : \{0,1\} \mapsto \{0,1\}$ e due qubits negli stati generici $|x\rangle$ e $|y\rangle$, che fungono da primo e secondo registro in un circuito quantistico. Si suppone anche di poter creare, disponendo degli opportuni gates, una matrice unitaria U_f tale che la sua azione sullo stato del sistema sia $|x, y\rangle = |x\rangle\,|y\rangle \overset{U_f}{\to} |x\rangle\,|y \oplus f(x)\rangle$. Nel caso in cui $|y\rangle = |0\rangle$, si ha $|x\rangle\,|0\rangle \overset{U_f}{\to} |x\rangle\,|f(x)\rangle$, ossia la funzione f è stata valutata in x nel secondo registro. Si considera ora il semplice circuito quantistico:



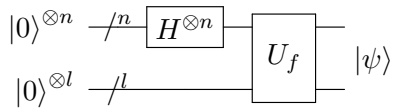
La sua azione sullo stato iniziale é:

$$|0\rangle\,|0\rangle \overset{H}{\to} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\,|0\rangle = \frac{1}{\sqrt{2}}(|0,0\rangle + |1,0\rangle) \overset{U_f}{\to} \frac{1}{\sqrt{2}}\big(\,|0, f(0)\rangle + |1, f(1)\rangle\,\big)$$

$$\implies |\psi\rangle = \frac{|0, f(0)\rangle + |1, f(1)\rangle}{\sqrt{2}} \quad .$$

Con una sola applicazione del circuito, f viene quindi valutata contemporaneamente in 0 e 1 e questa informazione risiede nella sovrapposizione dello

stato $|\psi\rangle$, anche se non direttamente accessibile all'osservatore. Il procedimento si può facilmente generalizzare per una funzione $f : \{0,1\}^n \mapsto \{0,1\}^l$, considerando il seguente circuito quantistico:



Lo stato iniziale del sistema è dunque $|0\rangle^{\otimes n} \otimes |0\rangle^{\otimes l} = |0\rangle\,|0\rangle$ in forma decimale. Applicando quindi il gate di Hadamard sul primo registro si ottiene una sovrapposizione quantistica dei 2^n stati della base computazionale di $\mathbb{C}^{2^n}$

$$\left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right)^{\otimes n} \otimes |0\rangle = \left(\frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle\right) \otimes |0\rangle \tag{1.4.1}$$

con $x = x_1 2^{n-1} + x_2 2^{n-2} + \ldots + x_n 2^0$ e $x_1, \ldots, x_n \in \{0,1\}$. Applicando infine il gate U_f si ottiene:

$$\frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle \otimes |0 \oplus f(x)\rangle = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle\,|f(x)\rangle = |\psi\rangle\,. \tag{1.4.2}$$

L'ultima equazione indica quindi che lo stato finale del circuito è una sovrapposizione quantistica di stati dove la funzione f è stata valutata per tutti i valori di x fra 0 e $2^n - 1$. Come prima, questa informazione su f non è direttamente utile e accessibile all'osservatore, in quanto un procedimento di misura farebbe collassare lo stato $|\psi\rangle$ del sistema in uno stato particolare $|\overline{x}\rangle\,|f(\overline{x})\rangle$ con $0 \leq \overline{x} \leq 2^n - 1$ e probabiltà $1/2^n$.

1.5 La trasformata di Fourier quantistica

La trasformata di Fourier quantistica (QFT) è l'operazione alla base di molti algoritmi quantistici, fra cui l'algoritmo di Shor. Si considera, in generale, la base computazionale in notazione decimale $\{|x\rangle\}_{x=0,\ldots,N-1}$ di $\mathbb{C}^N$. La QFT è definita come l'operatore lineare che agisce sulla base ortonormale considerata nel modo seguente:

$$|x\rangle \to QFT(|x\rangle) \equiv |\hat{x}\rangle = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{2\pi i x y/N}\,|y\rangle = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} \omega^{xy}\,|y\rangle \tag{1.5.1}$$

con $\omega = e^{2\pi i/N}$. L'azione di questo operatore su un generico stato di $\mathbb{C}^N$ è dunque:

$$|\psi\rangle = \sum_{x=0}^{N-1} \alpha_x\,|x\rangle \to \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} \alpha_x \sum_{y=0}^{N-1} e^{2\pi i x y/N}\,|y\rangle = \sum_{y=0}^{N-1} C_y\,|y\rangle\,, \tag{1.5.2}$$

$$\text{con} \quad C_y = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} \alpha_x e^{2\pi i x y/N}.$$

Come visto in precedenza, quando si lavora con un registro quantistico di n qubits $|x_1\rangle, \ldots, |x_n\rangle$, si pone $N = 2^n$ in quanto lo stato totale del sistema appartiene a $\mathbb{C}^{2^n}$. Vale la pena sottolineare che la QFT è una trasformazione che mappa $\mathbb{C}^{\mathbb{N}}$ in $\mathbb{C}^{\mathbb{N}}$ ed opera un cambiamento di base, da quella computazionale a quella di Fourier. Nel semplice caso di un solo qubit, in cui $|x\rangle = |x_1\rangle$, si ha:

$$|x_1\rangle \to \frac{1}{\sqrt{2}} \sum_{y=0}^{2-1} e^{2\pi i x_1 y/2} |y\rangle = \frac{1}{\sqrt{2}} \left(|0\rangle + e^{\pi i x_1} |1\rangle\right) = \frac{1}{\sqrt{2}} \left(|0\rangle \pm |1\rangle\right) = |\pm\rangle,$$

equivalente all'azione del gate di Hadamard, che manda appunto la base computazionale $\{|0\rangle, |1\rangle\}$ nella base di Fourier $\{|\pm\rangle\}$. Ricordando quanto detto sulla notazione per la forma decimale (1.1.3), si può riscrivere la QFT con alcuni passaggi algebrici:

$$|x\rangle \to \frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} e^{2\pi i x y/2^n} |y\rangle \tag{1.5.3}$$

$$\implies |x_1 \ldots x_n\rangle \to \frac{1}{\sqrt{2^n}} \sum_{y_1=0}^{1} \cdots \sum_{y_n=0}^{1} e^{2\pi i x \left(\sum_{l=1}^{n} y_l 2^{-l}\right)} |y_1 \ldots y_n\rangle \tag{1.5.4}$$

$$= \frac{1}{\sqrt{2^n}} \sum_{y_1=0}^{1} \cdots \sum_{y_n=0}^{1} \bigotimes_{l=1}^{n} e^{2\pi i x y_l 2^{-l}} |y_l\rangle \tag{1.5.5}$$

$$= \frac{1}{\sqrt{2^n}} \bigotimes_{l=1}^{n} \left[\sum_{y_l=0}^{1} e^{2\pi i x y_l 2^{-l}} |y_l\rangle\right] \tag{1.5.6}$$

$$= \frac{1}{\sqrt{2^n}} \bigotimes_{l=1}^{n} \left[|0\rangle + e^{2\pi i x 2^{-l}} |1\rangle\right] \tag{1.5.7}$$

$$= \frac{\left(|0\rangle + e^{2\pi i x/2} |1\rangle\right) \otimes \cdots \otimes \left(|0\rangle + e^{2\pi i x/2^n} |1\rangle\right)}{\sqrt{2^n}} \quad . \tag{1.5.8}$$

Nella rappresentazione in forma binaria si può anche introdurre l'utile notazione:

$$0.x_l x_{l+1} \ldots x_m = \frac{x_l}{2} + \frac{x_{l+1}}{2^2} + \ldots \frac{x_m}{2^{m-l+1}} \quad . \tag{1.5.9}$$

In particolare quindi: $0.x_1 \ldots x_n = x_1/2 + \ldots x_n/2^n$. Usando la (1.5.9) e considerando la periodicità dell'esponenziale complesso si può scrivere la trasformata di Fourier quantistica nella forma:

$$QFT(|x_1 \ldots x_n\rangle) = \frac{\left(|0\rangle + e^{2\pi i 0.x_n} |1\rangle\right) \otimes \cdots \otimes \left(|0\rangle + e^{2\pi i 0.x_1 x_2 \ldots x_n} |1\rangle\right)}{\sqrt{2^n}}. \tag{1.5.10}$$

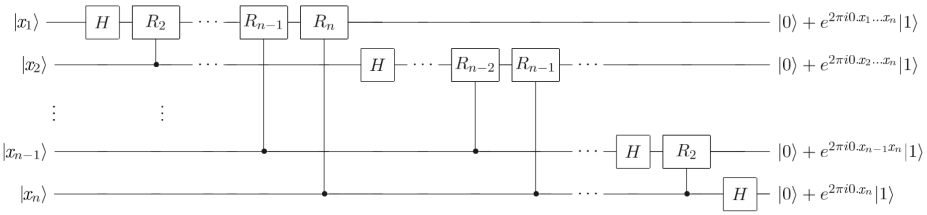


Figura 1.5.1: Circuito quantistico che implementa la QFT. Omesso per chiarezza il fattore di normalizzazione degli stati finali. Per ottenere lo stato dell'equazione (1.5.10) bisognerebbe aggiungere degli $SWAP$-gates fra qubits adiacenti.

L'espressione (1.5.10) è molto utile poiché permette di costruire un circuito quantistico, con i gates visti nel paragrafo 1.2, che implementi efficacemente la trasformata di Fourier quantistica. Tale circuito è rappresentato in figura 2.4.3. Lo stato iniziale del sistema è $|x_1 \dots x_n\rangle$, in rappresentazione binaria. Seguendo il circuito rappresentato, si considera il primo qubit e si applica ad esso il gate di Hadamard [1.5]. Lo stato risultate è:

$$\frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i x_1/2}\,|1\rangle\right)|x_2 \dots x_n\rangle\,.$$

Si applica successivamente il gate Controlled-R_2 sempre al primo qubit, con target il secondo qubit $|x_2\rangle$, ottenendo:

$$\frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i x_1/2} e^{2\pi i x_2/2^2}\,|1\rangle\right)|x_2 \dots x_n\rangle =$$

$$\frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i (x_1/2 + x_2/4)}\,|1\rangle\right)|x_2 \dots x_n\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i 0.x_1 x_2}\,|1\rangle\right)|x_2 \dots x_n\rangle\,.$$

Si itera dunque questo procedimento applicando al primo qubit tutti i gates Controlled-R_k, fino al Controlled-R_n, ottenendo come stato:

$$\frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i 0.x_1 x_2 \dots x_n}\,|1\rangle\right)|x_2 \dots x_n\rangle\,.$$

Si passa poi al secondo qubit, e si applica lo stesso procedimento usato per il primo, applicando quindi prima il gate H e poi i gates CR_k fino a CR_{n-1}. Lo stato risultate è a questo punto:

$$\frac{1}{\sqrt{2^2}}\left(|0\rangle + e^{2\pi i 0.x_1 x_2 \dots x_n}\,|1\rangle\right)\left(|0\rangle + e^{2\pi i 0.x_2 \dots x_n}\,|1\rangle\right)|x_3 \dots x_n\rangle\,.$$

Iterando nuovamente questo procedimento per tutti gli n qubits si ottiene come stato finale:

$$\frac{1}{\sqrt{2^n}}\left(|0\rangle + e^{2\pi i 0.x_1 x_2 \dots x_n}\,|1\rangle\right)\left(|0\rangle + e^{2\pi i 0.x_2 \dots x_n}\,|1\rangle\right)\cdots\left(|0\rangle + e^{2\pi i 0.x_n}\,|1\rangle\right),$$

che coincide con l'espressione (1.5.10), a meno di uno scambio dell'ordine dei qubits. Il circuito in figura 2.4.3 implementa quindi la QFT, e ciò dimostra che è una trasformazione unitaria. Per realizzare tale circuito, sono stati impiegati n gates per il primo qubit (il gate H più $n-1$ gates CR_k), $n-1$ gates per il secondo qubit (il gate H più $n-2$ gates CR_k), e così via per tutti gli n qubits. Il numero totale di gates usati è quindi:

$$n + (n-1) + \cdots + 1 = \sum_{i=1}^{n} i = \frac{n(n+1)}{2} = \frac{n^2}{2} + \frac{n}{2} \quad .$$

A questi si devono aggiungere i gates per scambiare i qubits: avendo utilizzato n qubits, sono necessari al più $n/2$ scambi, ciascuno dei quali può essere realizzato usando 3 $CNOT$ gates, per un totale di $3n/2$ gates. Si può quindi implementare efficacemente la QFT utilizzando $n^2/2 + 2n \implies O(n^2)$ gates quantistici. Utilizzando un algoritmo classico, invece, per fare la trasformata di Fourier discreta di un insieme di 2^n elementi sono necessari $O(n2^n)$ gates. Questo significa che un computer quantistico necessita di un numero di operazioni esponenzialmente minore (e quindi un tempo di esecuzione esponenzialmente minore) rispetto ad un computer classico per svolgere questo compito.

Capitolo 2

L'algoritmo di fattorizzazione di Shor

Come è noto fin dall'antichità, un numero intero positivo $N \in \mathbb{N}$ si può scrivere come prodotto di numeri primi, nella forma seguente:

$$N = p_1^{a_1} p_2^{a_2} \dots p_k^{a_k} \quad . \tag{2.0.1}$$

Il Teorema fondamentale dell'Aritmetica [8] afferma che tale rappresentazione, detta fattorizzazione di N, esiste sempre ed è unica. Mentre verificare che una fattorizzazione corrisponde ad un dato numero intero N è un problema computazionalmente molto facile (è sufficiente eseguire le moltiplicazioni), il problema inverso è invece molto complesso e dispendioso, dal punto di vista della computazione classica. Fattorizzare un numero intero è infatti un problema che appartiene alla classe computazionale **NP**, ossia l'insieme di quei problemi per cui non è noto un algoritmo efficiente per risolverli, ma di cui è facile verificarne una soluzione, qualora trovata. La classe computazionale **P** racchiude invece quei problemi risolvibili in maniera efficiente da un computer classico. Stabilire se queste classi computazionali coincidano o meno è ancora un problema aperto. Nella teoria della complessità computazionale, un algoritmo è detto efficiente se riesce a trovare una soluzione ad un dato problema in un tempo polinomiale nelle dimensioni del problema stesso, ossia eseguendo un numero di operazioni con andamento polinomiale rispetto alla dimensione del problema. Attualmente, non è ancora noto un algoritmo classico che riesca a fattorizzare un numero intero in maniera efficiente. Infatti, l'algoritmo classico computazionalmente migliore[1], allo stato attuale delle cose, necessita di $O(\exp(cL^{1/3}\log^{2/3} L))$ [1, 9] per fattorizzare un numero N di $L = \lceil \log_2 N \rceil$ bits. Il numero di operazioni, e quindi il tempo di esecuzione, cresce esponenzialmente in base al valore di N. Ed è proprio l'intrattabilità computazionale di questo problema il fondamento degli attuali sistemi di crittografia a chiave pubblica, come l'algoritmo RSA

[1] The general Number Field Sieve [9]

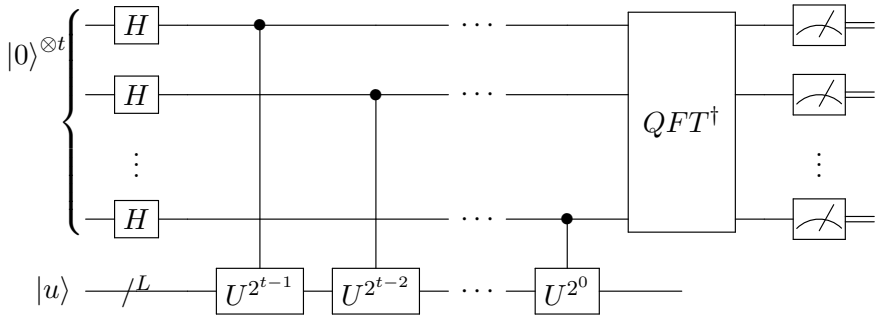
[3.2], che è il più largamente utilizzato. Il problema della fattorizzazione efficiente di un numero intero acquista quindi un ulteriore livello di importanza, poiché su di esso si basano gli attuali sistemi di sicurezza informatica. Non stupisce dunque che il lavoro pubblicato da Peter Shor nel 1994 [5] abbia portato un notevole interesse nell'ambito della computazione quantistica. In tale articolo, come verrà spiegato più avanti, viene presentato un algoritmo in grado di fattorizzare un numero intero in un tempo polinomiale in N, se eseguito su un computer quantistico. L'algoritmo di Shor è in grado infatti di fattorizzare un numero intero N di L bit in $O(L^2 \log L \log\log L)$ operazioni [1], esponenzialmente più veloce di un algoritmo classico. Nel seguito si analizzeranno tutti i passaggi che conducono a questo fondamentale risultato.

2.1 Quantum Phase Estimation

Un'applicazione interessante della QFT trattata nel precedente capitolo è l'algoritmo *Quantum Phase Estimation* (QPE), usato frequentemente come subroutine in molti algoritmi quantistici, fra cui l'algoritmo di Shor. Si considera un operatore unitario $U : \mathbb{C}^{2^L} \mapsto \mathbb{C}^{2^L}$ con $L \in \mathbb{N}$. Si può quindi interpretare U come un generico gate su L qubits. Si suppone inoltre di conoscere un autovettore $|u\rangle$ di U, da cui:

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

con $\varphi_u \in [0, 1)$ e non nota. Lo scopo della QPE è quello di determinare la fase φ_u relativa all'autoket $|u\rangle$. A tale scopo si considera un circuito quantistico a due registri, $|Reg1\rangle$ e $|Reg2\rangle$, il primo contenente t qubits e il secondo L qubits, ossia i qubits necessari per esprimere $|u\rangle$. Il numero t dei qubits del primo registro dipende invece dalla precisione con cui si vuole determinare il valore di φ_u, e dalla probabilità di successo dell'algoritmo[2]. Si considera il seguente circuito quantistico:



[2]Tale questione verrà chiarita meglio nel seguito.

Si osserva anzitutto come agiscono i gates Controlled-U^{2^j} (CU^{2^j}):

$$CU^{2^j}\left[(|0\rangle+|1\rangle)\,|u\rangle\right] = CU^{2^j}(|0\rangle\,|u\rangle) + CU^{2^j}(|1\rangle\,|u\rangle) = |0\rangle\,|u\rangle + |1\rangle\,U^{2^j}\,|u\rangle$$
$$= |0\rangle\,|u\rangle + |1\rangle\,e^{2\pi i\varphi_u 2^j}\,|u\rangle = (|0\rangle + e^{2\pi i\varphi_u 2^j}\,|1\rangle)\,|u\rangle\,.$$

Lo stato iniziale del circuito è $|0\rangle^{\otimes t}\,|u\rangle$. Dopo l'applicazione dei gates di Hadamard al primo registro e dei gates CU^{2^j} si ottiene:

$$|Reg1\rangle\,|Reg2\rangle = |0\rangle^{\otimes t}\,|u\rangle \xrightarrow{H} \frac{1}{\sqrt{2^t}}\sum_{x=0}^{2^t-1}|x\rangle\,|u\rangle = \left(\frac{|0\rangle+|1\rangle}{\sqrt{2}}\right)^{\otimes t}|u\rangle \xrightarrow{CU^{2^j}}$$

$$\to \frac{1}{\sqrt{2^t}}\underbrace{\left(|0\rangle+e^{2\pi i 2^{t-1}\varphi_u}|1\rangle\right)}_{\text{primo qubit}}\otimes\cdots\otimes\underbrace{\left(|0\rangle+e^{2\pi i 2^1\varphi_u}|1\rangle\right)}_{(t-1)\text{-esimo qubit}}\otimes\underbrace{\left(|0\rangle+e^{2\pi i 2^0\varphi_u}|1\rangle\right)}_{t\text{-esimo qubit}}|u\rangle$$

Quest'ultimo stato si può scrivere anche in maniera più compatta, sfruttando la relazione (1.5.3):

$$\frac{1}{\sqrt{2^t}}\sum_{x=0}^{2^t-1}e^{2\pi i\varphi_u x}\,|x\rangle\,|u\rangle = \frac{1}{\sqrt{2^t}}\sum_{x=0}^{2^t-1}|x\rangle\,U^x\,|u\rangle\,. \tag{2.1.1}$$

Siccome la fase $\varphi_u \in [0,1)$, si può scrivere in forma binaria come:[3]

$$\varphi_u = 0.\varphi_1\varphi_2\ldots\varphi_t \implies \varphi_1\varphi_2\ldots\varphi_t = 2^t\varphi_u$$

dove $\varphi_1,\ldots,\varphi_t \in \{0,1\}$ rappresentano i t bits che si suppone siano sufficienti per esprimere esattamente φ_u. Premesso ciò, e ricordando la notazione (1.5.9), si vede che lo stato del primo registro calcolato sopra si può scrivere come:

$$\frac{1}{\sqrt{2^t}}\left(|0\rangle+e^{2\pi i 0.\varphi_t}\,|1\rangle\right)\left(|0\rangle+e^{2\pi i 0.\varphi_{t-1}\varphi_t}\,|1\rangle\right)\ldots\left(|0\rangle+e^{2\pi i 0.\varphi_1\ldots\varphi_t}\,|1\rangle\right)$$

che coincide esattamente con l'espressione della Trasformata di Fourier quantistica (1.5.10) per lo stato $|\varphi_1\ldots\varphi_t\rangle = |2^t\varphi_u\rangle$. Lo stato espresso dall'equazione (2.1.1) è quindi: $|Reg1\rangle\,|Reg2\rangle = QFT(|\varphi_1\ldots\varphi_t\rangle)\,|u\rangle$. A questo punto, sempre seguento lo schema del circuito quantistico raffigurato, si applica la Trasformata di Fourier quantistica inversa ($QFT^\dagger$) al primo registro, ottenendo infine lo stato: $|\varphi_1\ldots\varphi_t\rangle\,|u\rangle$. Effettuando una misura del primo registro si possono leggere i t bit $\varphi_1,\ldots,\varphi_t$ e ricostruire il valore della fase φ_u. Tuttavia, φ_u potrebbe richiedere più di t bits (e quindi più di t

[3] In questo caso si è supposto di poter esprimere φ_u in maniera esatta usando t bits, equivalenti al numero di qubits usati nel primo registro. Ciò equivale a ipotizzare che $2^t\varphi_u$ sia un numero intero.

qubits nel primo registro) per essere espressa esattamente, e potrebbe anche non essere un numero razionale. In questi casi si può dimostrare che l'applicazione della $QFT^\dagger$ produce la miglior stima di φ_u ottenibile con t bits con probabilità almeno $4/\pi^2 \approx 0.405$ [13]. Per vederlo, supponiamo che $0.\varphi_1 \cdots \varphi_t = \tilde{\varphi_u} = \tilde{\phi_u}/2^t$ sia la miglior stima di φ_u ottenibile con t bits. $\tilde{\phi_u}$ rappresenta quindi l'intero più vicino a $2^t\varphi_u$ e si può quindi porre $\varphi_u = \tilde{\phi_u}/2^t + \delta = \tilde{\varphi_u} + \delta \implies 0 < |\delta| \leq \frac{1}{2^{t+1}}$. Applicando la $QFT^\dagger$ al primo registro dello stato nell'equazione (2.1.1) si ottiene:

$$QFT^\dagger \left(\frac{1}{\sqrt{2^t}} \sum_{x=0}^{2^t-1} e^{2\pi i \varphi_u x} |x\rangle \right) = \frac{1}{2^t} \sum_{y=0}^{2^t-1} \sum_{x=0}^{2^t-1} e^{\frac{-2\pi i x y}{2^t}} e^{2\pi i \varphi_u x} |y\rangle$$
$$= \frac{1}{2^t} \sum_{y=0}^{2^t-1} \sum_{x=0}^{2^t-1} e^{\frac{-2\pi i x y}{2^t}} e^{2\pi i (\frac{\tilde{\phi_u}}{2^t} + \delta) x} |y\rangle = \frac{1}{2^t} \sum_{y=0}^{2^t-1} \sum_{x=0}^{2^t-1} e^{\frac{2\pi i (\tilde{\phi_u} - y) x}{2^t}} e^{2\pi i \delta x} |y\rangle$$

Il coefficiente di Fourier dello stato $|y\rangle = |\tilde{\phi_u}\rangle = |\varphi_1 \ldots \varphi_t\rangle$ è quindi dato dalla serie geometrica:

$$\frac{1}{2^t} \sum_{x=0}^{2^t-1} \left(e^{2\pi i \delta} \right)^x = \frac{1}{2^t} \left(\frac{1 - \left(e^{2\pi i \delta} \right)^{2^t}}{1 - e^{2\pi i \delta}} \right). \tag{2.1.2}$$

Siccome $|\delta| \leq \frac{1}{2^{t+1}}$, segue che $2\pi\delta 2^t \leq \pi$, e quindi $|1 - e^{2\pi i \delta 2^t}| \geq \frac{2\pi\delta 2^t}{\pi/2} = 4\delta 2^t$. Vale inoltre la relazione $|1 - e^{2\pi i \delta}| \leq 2\pi\delta$. La probabilità di osservare lo stato $|\varphi_1 \ldots \varphi_t\rangle$ è data dal modulo quadro del corrispondente coefficiente di Fourier, per cui si ha:

$$\left| \frac{1}{2^t} \left(\frac{1 - e^{2\pi i \delta 2^t}}{1 - e^{2\pi i \delta}} \right) \right|^2 \geq \left(\frac{1}{2^t} \left(\frac{4\delta 2^t}{2\pi\delta} \right) \right)^2 = \frac{4}{\pi^2} \approx 0.405 \quad . \tag{2.1.3}$$

La probabilità di successo dell'algoritmo può essere ulteriormente aumentata fino a $1 - \epsilon$ con $\epsilon > 0$, prendendo un numero di qubits nel primo registro pari a $t = n + O(\log_2(1/\epsilon)) = n + \lceil \log_2(2 + 1/2\epsilon) \rceil$ per ottenere la miglior stima della fase φ_u disponibile con n bits [1, 13].

2.2 Order-finding

Una delle applicazioni più interessanti della QPE è trovare l'ordine, o il periodo, di una determinata funzione. Questo problema è strettamente legato al problema della fattorizzazione di un numero intero e costituirà un passaggio importante nell'algoritmo quantistico di Shor. Si considerano due interi positivi $m, N \in \mathbb{N}$ con $1 < m < N$ e coprimi, ossia privi di fattori

comuni, tranne chiaramente 1. Sia $L = \lceil \log_2 N \rceil$[4] il numero di bit necessari per esprimere N. Si considera ora la seguente funzione, detta esponenziale modulare

$$f_{m,N}(x) = m^x \pmod N \ ,$$

che fornisce il resto della divisione fra m^x ed N. Si definisce l'ordine r della funzione $f_{m,N}$ come il più piccolo intero tale per cui $f_{m,N}(r) = m^r \pmod N = 1$. Questo equivale a scrivere $m^r \equiv 1 \pmod N$ nell'aritmetica modulo N. L'ordine r rappresenta anche il periodo della funzione $f_{m,N}$, infatti:

$$\begin{aligned} m^r &= kN + 1 \\ m^{r+1} &= kNm + m \\ m^{r+1} \pmod N &= m \pmod N \\ m^{r+x} \pmod N &= m^x \pmod N \\ f_{m,N}(x+r) &= f_{m,N}(x) \ . \end{aligned}$$

Si tratta dunque di una funzione periodica, di periodo r. Da ciò segue anche che $r \leq N$. Il problema dell'*Order-Finding* consiste proprio nel determinare l'ordine r in maniera efficiente, ossia svolgendo un numero di operazioni polinomialmente dipendente da N. L'idea di Peter Shor per risolvere questo problema fu di applicare la QPE, vista precedentemente, prendendo come gate quantistico[5]

$$U_{m,N} \left|x\right\rangle = \left|mx \pmod N\right\rangle \ ,$$

con $\left|x\right\rangle \in \{\left|0\right\rangle, \ldots, \left|N-1\right\rangle\}$, base computazionale di $\mathbb{C}^N$ espressa in forma decimale. Siccome $mx \pmod N$ rappresenta il resto della divisione euclidea fra mx ed N, ne segue che $mx \pmod N < N$, e quindi anche lo stato $\left|mx \pmod N\right\rangle$ appartiene alla base computazionale di $\mathbb{C}^N$. Per rendere effettivamente $U_{m,N}$ un gate su L qubits, come richiesto nella QPE, bisogna quindi prendere $L = \lceil \log_2 N \rceil$, in modo tale che $\mathbb{C}^N \longrightarrow \mathbb{C}^{2^L}$. Per applicare la QPE serve trovare anche uno o più autostati del gate $U_{m,N}$. Si procede dunque come segue, ricordando che $m < N$:

$$\begin{aligned} &U_{m,N} \left|1\right\rangle = \left|m \pmod N\right\rangle = \left|m\right\rangle \\ &U_{m,N}^2 \left|1\right\rangle = U_{m,N} \left|m\right\rangle = \left|m^2 \pmod N\right\rangle \\ &\vdots \\ &U_{m,N}^r \left|1\right\rangle = \left|m^r \pmod N\right\rangle = \left|1\right\rangle \end{aligned}$$

[4] Deve essere infatti $N < 2^L \implies L > \log_2 N$. Si prende quindi $L = \lceil \log_2 N \rceil$ che restituisce il più piccolo intero $\geq \log_2 N$.

[5] Notare che il gate $U_{m,N}$ così definito è un gate unitario, in quanto m ed N sono coprimi.

Dopo r applicazioni del gate $U_{m,N}$ si torna allo stato $|1\rangle$[6] di partenza. Si potrebbe quindi prendere come autostato di $U_{m,N}$ una combinazione lineare normalizzata dei precedenti vettori, e definire:

$$|u\rangle = \frac{1}{\sqrt{r}} \sum_{x=0}^{r-1} |m^x \pmod N\rangle$$

Si ha infatti:

$$\begin{aligned} U_{m,N}|u\rangle &= \frac{1}{\sqrt{r}} \sum_{x=0}^{r-1} U_{m,N}|m^x \pmod N\rangle = \frac{1}{\sqrt{r}} \sum_{x=0}^{r-1} |m^{x+1} \pmod N\rangle \\ &= \frac{1}{\sqrt{r}} \sum_{x=1}^{r-1} |m^x \pmod N\rangle + |1\rangle = \frac{1}{\sqrt{r}} \sum_{x=0}^{r-1} |m^x \pmod N\rangle = |u\rangle \end{aligned}$$

Tuttavia questo autovettore non è di grande interesse poiché ha autovalore 1, e non compare esplicitamente l'ordine r. Si definisce dunque un set di vettori in cui nella combinazione lineare compaia anche una fase:

$$|u_s\rangle = \frac{1}{\sqrt{r}} \sum_{x=0}^{r-1} e^{-2\pi i s x/r} |m^x \pmod N\rangle \tag{2.2.1}$$

con $0 \leq s \leq r-1$. É facile vedere che:

$$U_{m,N}|u_s\rangle = e^{2\pi i s/r}|u_s\rangle = e^{2\pi i \varphi_s}|u_s\rangle \tag{2.2.2}$$

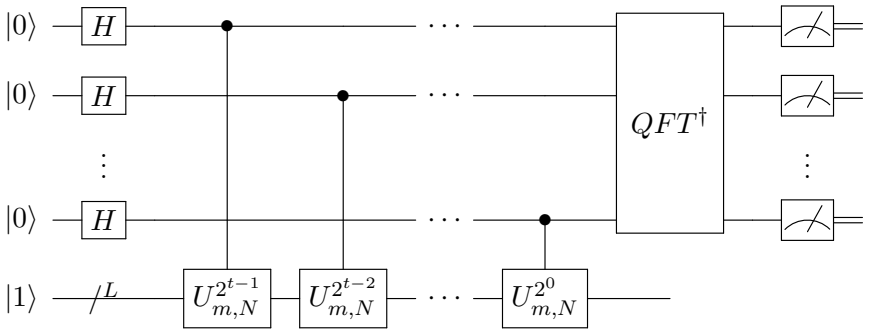
con $\varphi_s = s/r < 1$. Gli autovettori $\{|u_s\rangle\}_s$ formano anche un set di r vettori ortonormali. In quanto relativi al gate $U_{m,N}$, possono essere espressi come prodotto tensore di $\lceil \log_2 N \rceil = L$ qubits, posti nel secondo registro del circuito della QPE. Gli autostati $|u_s\rangle$ non possono però essere creati direttamente in quanto ciò presumerebbe la conoscenza dell'ordine r, che invece è lo scopo dell'algoritmo. Si può però utilizzare la seguente uguaglianza:

$$\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |u_s\rangle = |1\rangle \tag{2.2.3}$$

e considerare lo stato iniziale del secondo registro della QPE, $|Reg2\rangle = |1\rangle$, come una combinazione lineare degli autostati $|u_s\rangle$ di $U_{m,N}$. Si può ora applicare la QPE come visto nel paragrafo precedente, considerando due registri quantistici $|Reg1\rangle$ e $|Reg2\rangle$ di t e L qubits e il seguente circuito

[6]Notare che in questo caso si sta utilizzando la notazione decimale. Lo stato $|1\rangle$ non indica che il sistema è a un solo qubit, ma potrebbe essere la rappresentazione decimale di uno stato $|1\rangle = |00\ldots1\rangle = |0\rangle^{\otimes L-1}|1\rangle$ di L qubits.

quantistico:



Applicando il gate $H^{\otimes t}$ al primo registro e i gates Controlled-$U^{2^j}_{m,N}$ si ottiene, usando la relazione (2.1.1):

$$|0\rangle^{\otimes t}|1\rangle \xrightarrow{H^{\otimes t}} \frac{1}{2^{t/2}}\sum_{x=0}^{2^t-1}|x\rangle|1\rangle \xrightarrow{CU^{2^j}} \frac{1}{2^{t/2}}\sum_{x=0}^{2^t-1}|x\rangle U^x_{m,N}|1\rangle\,. \tag{2.2.4}$$

Si sfrutta ora l'equazione (2.2.3) per espandere lo stato iniziale del secondo registro, sapendo che gli $|u_s\rangle$ sono autostati di $U_{m,N}$:

$$\frac{1}{\sqrt{r2^t}}\sum_{x=0}^{2^t-1}\sum_{s=0}^{r-1}|x\rangle U^x_{m,N}|u_s\rangle = \frac{1}{\sqrt{r}}\sum_{s=0}^{r-1}\left(\frac{1}{\sqrt{2^t}}\sum_{x=0}^{2^t-1}e^{2\pi i x s/r}|x\rangle\right)|u_s\rangle\,. \tag{2.2.5}$$

Applicando la Trasformata di Fourier quantistica inversa ($QFT^\dagger$) al primo registro si ottiene la combinazione lineare di r vettori:

$$\frac{1}{\sqrt{r}}\sum_{s=0}^{r-1}|2^t\tilde{\varphi}_s\rangle|u_s\rangle = \frac{1}{\sqrt{r}}\sum_{s=0}^{r-1}|\varphi_1\ldots\varphi_t\rangle|u_s\rangle \tag{2.2.6}$$

dove $\varphi_s = s/r < 1$, e $\tilde{\varphi}_s$ è la stima della fase φ_s ottenuta con i t qubits utilizzati nel primo registro. Eseguendo una misura sul primo registro, lo stato finale del sistema risulta essere $|2^t\tilde{\varphi}_{\overline{s}}\rangle|u_{\overline{s}}\rangle$ con $0 \le \overline{s} < r$, da cui si può estrarre il valore di $\tilde{\varphi}_{\overline{s}}$, dove $\overline{s}$ è un particolare valore di s selezionato dal processo di misura. A questo punto si può estrarre il valore dell'ordine r calcolando la frazione minima più vicina a $\tilde{\varphi}_{\overline{s}}$. Qualora si scelga, per esempio, $t = 2L+1$ è verificata la condizione $|s/r - \tilde{\varphi}_s| \le 2^{-2L-1} = 1/(2\times 2^{2L}) \le 1/(2N^2) << 1$ per N sufficientemente grande. La probabilità che dal processo di misura venga selezionata una particolare fase $\tilde{\varphi}_{\overline{s}}$ è data da:

$$Prob(\tilde{\varphi}_{\overline{s}}) = \left|\frac{1}{\sqrt{r}}\sum_{s=0}^{r-1}\langle 2^t\tilde{\varphi}_{\overline{s}}\otimes u_{\overline{s}}|2^t\tilde{\varphi}_s\otimes u_s\rangle\right|^2 = \left|\frac{1}{\sqrt{r}}\sum_{s=0}^{r-1}\delta_{\overline{s}s}\right|^2 = \frac{1}{r}\,, \tag{2.2.7}$$

ed è quindi indipendente dal valore di $\overline{s}$. Ci si chiede ora quante risorse computazionali richiede questo algoritmo. Supponiamo di prendere $t = O(L)$

qubits nel primo registro, con $L = \lceil \log_2 N \rceil$. Vengono utilizzati $t = O(L)$ gates di Hadamard sul primo registro e la Trasformata di Fourier quantistica inversa richiede un totale di $O(L^2)$ gates [1.5]. Vi sono poi i gates Controlled-$U_{m,N}^{2^j}$ per $j = 0, \ldots, t-1$. Come già visto, questi agiscono sul secondo registro nel seguente modo $U_{m,N}^{2^j} |1\rangle = |m^{2^j} \cdot 1 \pmod N\rangle$. Sembra quindi che questo calcolo richieda risorse esponenziali in t, e quindi in L. Tuttavia esiste un algoritmo classico per calcolare gli esponenziali in aritmetica modulare in maniera efficiente, utilizzando $O(L^3)$ gates. Risultano quindi un totale di $O(L^3)$ gates per arrivare alla stima della fase. Risulta quindi evidente come l'algoritmo dell'*order-finding* richieda un numero di risorse polinomiale, e quindi un tempo di esecuzioni polinomiale, se eseguito su un computer quantistico.

2.3 L'algoritmo di Shor: riduzione all'order-finding

Ora che sono stati presentati tutti gli elementi necessari, si può procedere ad illustrare compiutamente l'algoritmo di Shor in tutti i suoi passaggi. Lo scopo dell'algoritmo è, dato un numeri intero $N \in \mathbb{N}$, trovarne almeno un fattore non banale, e reiterando l'algoritmo trovarne i fattori primi, ricordando l'espressione (2.0.1) per la fattorizzazione.

STEP 1: Verificare se N è pari, e in caso affermativo restituire il fattore 2. Scegliere casualmente un intero positivo $m < N$ e calcolare $\gcd(m, N)$ con l'algoritmo di Euclide[7]. Se $\gcd(m, N) \neq 1$ restituire il fattore $\gcd(m, N)$ e ripetere l'algoritmo con il numero $N/\gcd(m, N)$. Se $\gcd(m, N) = 1$, m e N sono coprimi e si procede al passo successivo.

STEP 2: Si definisce $f_{m,N}(x) = m^x \pmod N$ e se ne calcola l'ordine r seguendo il procedimento del precedente paragrafo. Si preparano i due registri quantistici $|Reg1\rangle, |Reg2\rangle$ con $t = 2L + 1 + O(\log_2(1/\epsilon))$ ed L qubits rispettivamente, dove $L = \lceil \log_2 N \rceil$. Questo è l'unico passaggio ove è richiesto un computer quantistico.

STEP 3: Se r è dispari tornare allo STEP 1 e procedere con un diverso m. Se r è pari, dalla definizione di ordine si ha:

$$\left(m^{r/2} + 1\right)\left(m^{r/2} - 1\right) = m^r - 1 \equiv 0 \pmod N$$

$$\left(m^{r/2} + 1\right)\left(m^{r/2} - 1\right) = kN, \quad k \in \mathbb{N} \ .$$

[7] L'algoritmo di Euclide è descritto nella sezione 4.1

Notare che $m^{r/2}-1$ non può essere un multiplo di N, per definizione di r [8]. Se $m^{r/2} \not\equiv -1 \pmod{N}$ neanche $m^{r/2}+1$ è multiplo di N e quindi almeno uno fra questi due termini contiene un fattore non banale di N. Se invece $m^{r/2} \equiv -1 \pmod{N}$, tornare allo STEP 1.

STEP 4: Si calcolano $\gcd(m^{r/2}-1, N)$ e $\gcd(m^{r/2}+1, N)$ con l'algoritmo di Euclide e si ottiene un fattore non banale di N.

Riguardo allo STEP 3, data una fattorizzazione di N come nell'equazione (2.0.1), si ha che [1]:

$$p(r \text{ pari e } m^{r/2} \not\equiv -1 \pmod{N}) \geq 1 - \frac{1}{2^k} \quad . \tag{2.3.1}$$

Dunque, se $N = pq$, con p e q numeri primi, scelto $m \in \mathbb{N}$ come nello STEP 1, la probabilità di trovare un ordine r di f_m, N per cui l'algoritmo va a buon fine è almeno $3/4$. Il caso $N = pq$ è particolarmente interessante dal punto di vista applicativo, in quanto legato all'algoritmo RSA. In questo caso, inoltre, per ridurre le dimensioni del circuito, e quindi i tempi di esecuzione dell'algoritmo e il rumore quantistico, si può scegliere un numero t di qubits tale per cui $N^2 \leq 2^t < 2N^2$ [3, 15].

2.4 Implementazione e simulazioni

In questa sezione vengono riportati i risultati di alcune simulazioni dell'algoritmo di Shor, per la fattorizzazione di piccoli numeri interi nella forma $N = pq$. L'algoritmo è stato implementato usando il linguaggio di programmazione Python, ed in particolare la libreria Qiskit [4]. Le simulazioni sono state eseguite su un computer classico (utilizzando in cloud le risorse messe a disposizione da IBM), usato per simulare il comportamento di un computer quantistico. Le relazioni (2.2.7) e (4.2.6), viste nel precedente paragrafo e nell'appendice, indicano che la probabilità di ottenere una fase $\tilde{\varphi} = y/2^t$ dal processo di misura del primo registro è:

$$Prob(y) \approx \begin{cases} 0 & ry \neq 0 \pmod{Q} \\ \frac{1}{r} & ry = 0 \pmod{Q} \end{cases} \tag{2.4.1}$$

Si riportano nelle pagine seguenti i grafici delle distribuzioni di probabilità delle misure del primo registro ottenute nelle simulazioni dell'algoritmo di Shor, in particolare della QPE. In tutti i casi sono stati utilizzati $L = \lceil \log_2(N) \rceil$ qubits nel secondo registro e $t = 2L$ qubits nel primo, in modo tale che $N^2 \leq 2^t < 2N^2$, come visto nel precedente paragrafo. I grafici, realizzati tramite gnuplot, riportano i valori di y sulle ascisse e la rispettiva probabilità $Prob(y)$ sulle ordinate.

[8] Si avrebbe infatti che $m^{r/2} - 1 \equiv 0 \pmod{N} \implies m^{r/2} \equiv 1 \pmod{N}$, che contraddice il fatto che r sia il più piccolo intero tale per cui $m^r \equiv 1 \pmod{N}$.

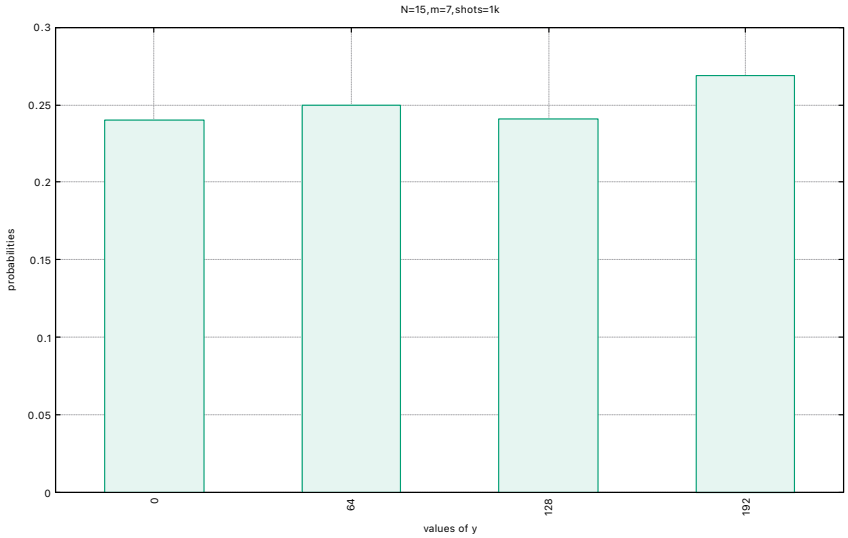


Figura 2.4.1: Sono state eseguite 1000 simulazioni con N=15 e m=7. In questo caso L=$\lceil \log_2(15) \rceil$=4 e $t = 2L$=8, per un totale di 12 qubits nei due registri. Si individuano 4 picchi ben definiti, di altezza $Prob(y) \approx 0.25$, a cui corrispondono le fasi $\tilde{\varphi} = y/2^t$=0, 0.25, 0.5, 0.75. L'algoritmo ha portato a termine efficacemente la fattorizzazione in circa il 50% dei casi.

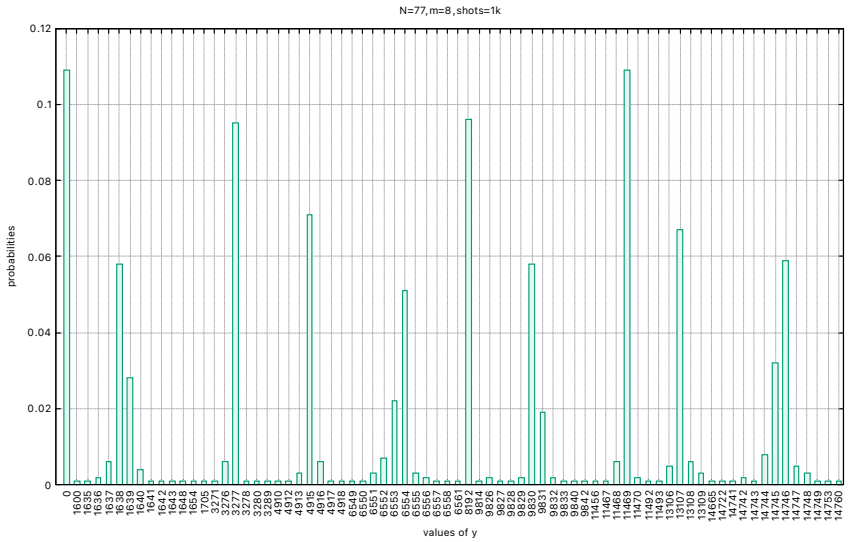


Figura 2.4.2: Il grafico è stato ottenuto eseguendo 1000 simulazioni con N=77 e m=8. In questo caso L=$\lceil \log_2(77) \rceil$=7 e t=14. Sono ben evidenti 10 picchi principali, da cui si può intuire che $r = 10$. Risultato confermato dall'algoritmo delle frazioni continue e dalla formula (4.2.6). L'algoritmo ha trovato i fattori primi di N=77 in circa il 41% dei casi.

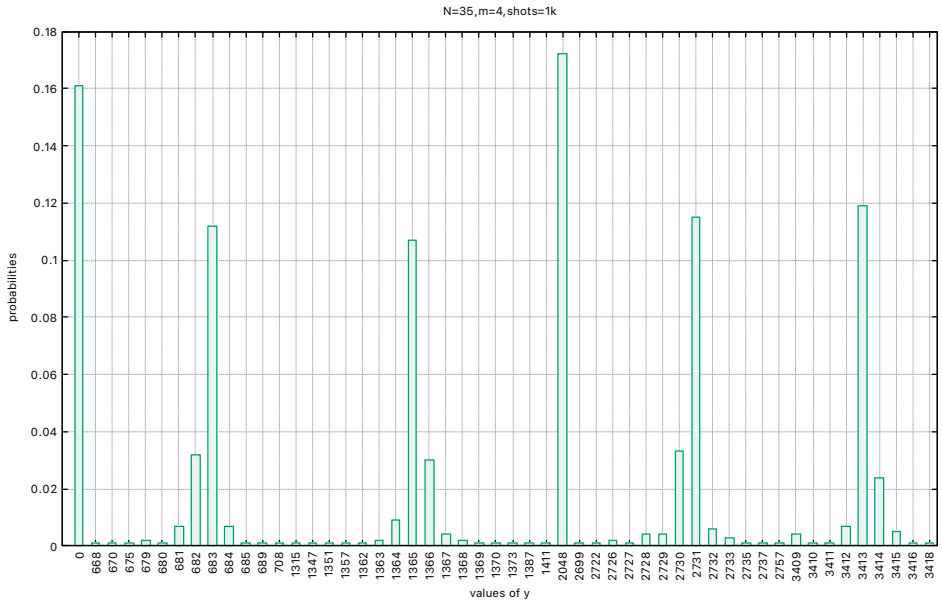


Figura 2.4.3: Il grafico è stato ottenuto eseguendo 1000 simulazioni con N=35 e m=4. Il circuito quantistico è stato realizzato usando L=$\lceil \log_2(35) \rceil$=6 qubits nel secondo registro e $t = 2L = 12$ qubits nel primo, sicché $Q = 2^t$=4096. Sono ben evidenti 6 picchi principali. Per esempio, il picco a y=683 (il primo dopo quello a y=0), corrisponde ad una fase $\tilde{\varphi} = y/2^t \approx 0{,}1667$. L'algoritmo delle frazioni continue fornisce facilmente l'ordine r=6. L'ordine della funzione $f(x) = 4^x \pmod{35}$ è infatti $r = 6$. I due picchi più alti, a y=0 e y=2048, hanno una probabilità rispettivamente di 0.161 e 0.171. Ciò è compatibile con la formula (4.2.6) poiché $1/r \approx 0.167$. Gli altri 4 picchi sono più bassi (corrispondono ad una probabilità ≈ 0.11), ma ciò è dovuto ai valori di y adiacenti aventi probabilità non nulla o trascurabile. L'algoritmo ha trovato i fattori primi di N=35 in circa il 33% dei casi.

Capitolo 3

Crittografia classica e quantistica

3.1 Introduzione

I primi sistemi crittografici classici ad essere stati utilizzati per la trasmissione di messaggi fra due parti sono quelli a chiave privata. Un semplice ma estremamente efficace esempio è il cifrario di Vernam, in cui due soggetti condividono una stessa chiave privata della stessa lunghezza del messaggio da trasmettere attraverso un canale potenzialmente violabile da terzi. Ad ogni lettera o carattere del messaggio viene associato un numero intero, oppure si usa una codifica in bits binari. Il messaggio viene quindi criptato e decriptato dalle due parti semplicemente sommando e sottraendo, usando l'aritmetica modulare[1], i bits della chiave a quelli del messaggio corrispondente. É stato dimostrato da C. Shannon nel 1949 [11] che il protocollo è sicuro, a patto di utilizzare la stessa chiave privata una sola volta. Vi sono però delle importanti criticità, come trovare un canale sicuro e fidato per scambiarsi i bits delle chiavi private, che lo rendono inapplicabile per un uso ampio e diffuso. Gli attuali sistemi di sicurezza informatica sono infatti basati su protocolli crittografici asimmetrici, o a chiave pubblica, così da evitare difficoltà nella distribuzione di un'unica chiave privata. In questi sistemi, ogni soggetto genera, tramite un particolare algoritmo, una chiave pubblica (P) e una chiave privata (S): la prima viene resa pubblica e utilizzata da un mittente per criptare i messaggi indirizzati verso il destinatario, la seconda rimane segreta ed è necessaria al destinatario per la decrittazione del messaggio. Gli algoritmi utilizzati devono garantire facilità nella generazione della coppia di chiavi, e affidare la sicurezza della chiave privata a problemi computazionali complessi privi di una soluzione efficiente nota. Il più largamente diffuso è l'algoritmo RSA, la cui efficacia è basata sulla difficoltà computazionale della

[1] Con una codifica in bit è sufficiente usare l'addizione modulo 2 per criptare e decriptare il messaggio, implementabile con uno XOR.

fattorizzazione su un computer classico. I numeri utilizzati negli attuali protocolli RSA hanno solitamente dalle 1024 alle 2048 cifre binarie (RSA-1024 ed RSA-2048), che corrispondono rispettivamente a 309 e 617 cifre decimali. Per quanto visto in precedenza, fattorizzare efficientemente tali numeri richiederebbe un computer quantistico con migliaia di qubits stabili, ad oggi non ancora disponibile. Nel seguito si vedrà come l'algoritmo quantistico di Shor sia potenzialmente in grado di violare un protocollo del genere e quali siano le implicazioni nella sicurezza informatica. Fortunatamente, la teoria dell'informazione quantistica offre delle soluzioni a questo problema, la cui sicurezza è fondata sulla correttezza delle leggi della meccanica quantistica. I protocolli di Crittografia quantistica, o scambio quantistico della chiave (QKD da *Quantum Key Distribution*) permettono a due soggetti di scambiarsi una chiave privata in maniera provatamente sicura, utilizzando qubits e un canale quantistico attraverso il quale scambiarli.

3.2 L'algoritmo RSA

L'algoritmo RSA [12] fu ideato nel 1977 da Rivest, Shamir e Adleman. La trattazione di questa parte presuppone una conoscenza di base della teoria dei numeri e dell'aritmetica modulare. La coppia di chiavi viene generata secondo i seguenti passaggi:

1. Scegliere casualmente due numeri primi p e q sufficientemente grandi[2].
2. Calcolare il prodotto $n = pq$.
3. Calcolare la funzione toziente di Eulero[3], $\phi(n) = \phi(pq) = (p-1)(q-1)$, e scegliere casualmente un numero intero $e < n$ che sia coprimo di $\phi(n)$, ossia $\gcd(e, \phi(n)) = 1$ [4].
4. Siccome e e $\phi(n)$ sono comprimi, si può calcolare l'inverso di e modulo $\phi(n)$: $d \equiv e^{-1} \pmod{\phi(n)} \implies ed \equiv 1 \pmod{\phi(n)}$.
5. La chiave pubblica è infine data da $P = (e, n)$. La chiave privata è invece $S = (d, n)$.

Si suppone ora di voler trasmettere un messaggio M, di al più $\lfloor \log_2 n \rfloor$ bits, tradotto in codice binario secondo uno schema di regole preciso (ad esempio ASCII). Se M ha più di $\lfloor \log_2 n \rfloor$ bits lo si divide in blocchi di al più $\lfloor \log_2 n \rfloor$

[2] In questo contesto significa almeno un migliaio di cifre binarie. Questi numeri possono essere trovati in maniera efficiente usando il Crivello di Eratostene.

[3] Sia $R(n) = \{r \in \mathbb{N} | 0 < r < n, \gcd(r, n) = 1\}$ l'insieme dei numeri interi coprimi e minori di n. La funzione di Eulero $\phi(n)$ è la cardinalità di questo insieme. Per un numero primo vale dunque $\phi(p) = p - 1$, e per un prodotto di numeri coprimi $\phi(pq) = \phi(p)\phi(q)$.

[4] gcd(greatest common divisor): massimo comun divisore.

bits. Usando la chiave pubblica del destinatario $P = (e, n)$ si esegue il processo di cifratura (E), che produce il crittogramma C:

$$C = E(M) = M^e \pmod{n} .$$

Una volta inviato C, il destinatario esegue il processo di decifrazione (D) usando la sua chiave privata $S = (d, n)$:

$$C \to D(C) = D(E(M)) = C^d \pmod{n} .$$

Alla fine della procedura, il destinatario deve riottenere il messaggio originale M, per cui si deve dimostrare che $D(E(M)) = D(C) = M$. A tal fine si ricorda che $ed \equiv 1 \pmod{\phi(n)}$, per cui, dalla definizione di modulo, $ed = 1 + k\phi(n)$ con $k \in \mathbb{N}$. Nella sezione 4.1 dell'appendice sono riportati alcuni teoremi di teoria dei numeri, fondamentali per il seguito della trattazione. Si divide la dimostrazione in due casi distinti:

1. M è coprimo sia di p che di q, e di conseguenza di n
2. M non è coprimo di $n \implies M$ è multiplo di p o q

Nel primo caso, siccome M e n sono coprimi, si può applicare il piccolo teorema di Fermat generalizzato: $M^{\phi(n)} \equiv 1 \pmod{n} \implies M^{k\phi(n)} = \left(M^{\phi(n)}\right)^k \equiv 1 \pmod{n}$, con $\phi(n) = (p-1)(q-1)$. Si ha allora:

$$\begin{aligned} D(C) = C^d \pmod{n} &= M^{ed} \pmod{n} && (3.2.1)\\ &= M^{1+k\phi(n)} \pmod{n} = MM^{k\phi(n)} \pmod{n} && (3.2.2)\\ &= M \cdot 1 \pmod{n} = M \pmod{n} = M. && (3.2.3) \end{aligned}$$

Nel secondo caso si suppone, senza perdita di generalità, che p divide M, da cui $M = kp, k \in \mathbb{N}$, mentre q non divide M. Da ciò segue che:

$$\begin{aligned} M = kp \equiv 0 \pmod{p} \quad \text{e} \quad M^{ed} = (kp)^{ed} \equiv 0 \pmod{p} \\ \implies M^{ed} \equiv M \pmod{p} . \end{aligned} \tag{3.2.4}$$

Siccome q non divide M, si può usare il piccolo teorema di Fermat:

$$\begin{aligned} M^{q-1} \equiv 1 \pmod{q} \implies M^{\phi(n)} = (M^{q-1})^{p-1} \equiv 1 \pmod{q} \\ \implies M^{ed} = MM^{k\phi(n)} \equiv M \cdot 1 = M \pmod{q} . \end{aligned} \tag{3.2.5}$$

Siccome p e q sono numeri primi, dalle equazioni (3.2.4) e (3.2.5) segue infine, per il teorema cinese del resto:

$$D(E(M)) = M^{ed} = M \pmod{pq} = M \pmod{n} .$$

In entrambi i casi il destinatario riesce a riottenere il messaggio iniziale. É immediato ora vedere come l'algoritmo quantistico di Shor possa violare il

protocollo RSA. Infatti, dalla chiave pubblica del destinatario $P = (e, n)$ si può dedurre n e, potendolo fattorizzare in maniera efficiente, dedurre p e q. A questo punto si può calcolare facilmente la funzione di Eulero $\phi(n)$ e quindi seguire l'algoritmo per la generazione delle chiavi per trovare la chiave privata $S = (d, n)$. Esiste anche un altro modo in cui si può violare l'algoritmo RSA, basato sull'*order-finding*. Supponiamo che Eva conosca il messaggio cifrato $C = M^e \pmod n$ e la chiave pubblica del destinatario (e, n), e sia in grado di calcolare l'ordine r di $(M^e)^r = 1 \pmod n$[5] tramite l'algoritmo dell'*order-finding*. Per il piccolo teorema di Fermat generalizzato, vale $(M^e)^{\phi(n)} = 1 \pmod n$ e ne segue che r divide $\phi(n)$. Siccome e e $\phi(n)$ sono coprimi per costruzione, ne segue che anche e ed r sono coprimi, e perciò si può calcolare l'inverso di e modulo r: $d' \equiv e^{-1} \pmod r \implies ed' = 1 \pmod r \implies ed' = 1 + k'r$. A questo punto Eva può ricostruire il messaggio originale in questa maniera:

$$C^{d'} = (M^e)^{d'} = MM^{k'r} = M \pmod n \ .$$

Tuttavia, come visto, non esiste un algoritmo classico efficiente per il calcolo dell'ordine. La forza dell'RSA si basa quindi sul fatto che è computazionalmente efficiente generare le chiavi e svolgere le operazioni di cifratura e decifratura, grazie all'esponenziale modulare, ma non sono noti algoritmi classici efficienti per la fattorizzazione e l'*order-finding*.

3.3 Il protocollo BB84

Nel 1984, H. Bennet e G. Brassard [10] descrissero nel loro articolo il primo protocollo di scambio quantistico della chiave (QKD). I principi alla base di questo protocollo crittografico sono quelli illustrati nel paragrafo 1.3, ossia il teorema No-Cloning e il fatto che non è possibile distinguere fra stati non ortogonali a meno di introdurre una perturbazione sul sistema. Lo scopo del protocollo è permettere a due soggetti, Bob e Alice, di creare in maniera sicura una chiave crittografica da utilizzare poi per criptare e decrittare un messaggio attraverso un canale pubblico con un sistema di crittografia a chiave privata, come un cifrario di Vernam. La comunicazione fra Bob e Alice avviene attraverso due canali: un canale classico pubblico e non modificabile[6], ed un canale quantistico per lo scambio dei qubits, che può essere potenzialmente disturbato dalla presenza di un terzo intercettatore, chiamato Eve. Sperimentalmente, il canale quantistico può essere realizzato con un cavo in fibra ottica nel quale far passare i fotoni, usati come qubits,

[5] In questo modo si suppone che M^e e n siano coprimi. Non si perde di generalità, in quando se avessero un fattore comune, p o q, questo potrebbe essere trovato con l'algoritmo di Euclide in maniera efficiente e poi procedere come nel caso precedente.

[6] I messaggi che Alice e Bob si scambiano su questo canale possono essere letti da terze parti ma non possono essere modificati o eliminati da nessuno.

generati da un dispositivo ottico capace di produrre fotoni polarizzati lungo quattro direzioni di polarizzazione. Nel seguito vengono illustrati tutti i passaggi del protocollo:

1: Alice genera casualmente due stringhe a e b di $4n$ bit ciascuna. Ella associa poi ai bit classici $\{0,1\}$ della stringa a, gli stati quantistici $\{|0\rangle, |1\rangle\}$ se il bit corrispondente della stringa b è 0, e $\{|+\rangle, |-\rangle\}$ se il bit corrispondente di b è 1. In questo modo Alice genera uno stato di $4n$ qubits,

$$|\psi\rangle = \bigotimes_{j=1}^{4n} |\psi_{a_j b_j}\rangle$$

dove a_j e b_j sono gli j-esimi bit delle stringhe a e b. Ogni qubit della stringa è dunque uno dei quattro stati:

$$|\psi_{00}\rangle = |0\rangle, \quad |\psi_{10}\rangle = |1\rangle$$

$$|\psi_{01}\rangle = |+\rangle = (|0\rangle + |1\rangle)/\sqrt{2}, \quad |\psi_{11}\rangle = |-\rangle = (|0\rangle - |1\rangle)/\sqrt{2}$$

Ciò significa che i bit di a vengono codificati nello stato $|\psi\rangle$ utilizzando la base computazionale (Z) oppure la base di Fourier (X), a seconda dei bit della stringa b, rispettivamente 0 e 1. Si può notare che i 4 stati quantistici non sono fra loro ortogonali e quindi non è possibile per un intercettatore distinguerli tutti con un procedimento di misura senza perturbare irrimediabilmente il sistema.

2: Alice manda lo stato risultate dei $4n$ qubits a Bob attraverso il canale quantistico pubblico, che pertanto può essere vittima di un intercettatore, Eve. Bob riceve quindi lo stato $\varepsilon|\psi\rangle$, dove ε rappresenta l'influenza sia del rumore ambientale del canale quantistico di trasmissione sia l'azione di un possibile intercettatore, e comunica l'avvenuta ricezione dei qubits.

3: A questo punto Bob genera una stringa b' di $4n$ bit casuali. Procede dunque a misurare i $4n$ qubits inviati da Alice utilizzando la base Z o X a seconda che il corrispettivo bit di b' sia 0 o 1. In questo modo, leggendo lo stato risultante, ottiene una stringa a' sempre di $4n$ bits.

4: Attraverso il canale classico pubblico Alice e Bob comunicano e confrontano i bits delle stringhe b e b', e quindi la loro scelta sulla base utilizzata per la codifica e la misura di ciascuno dei $4n$ qubits dello stato $|\psi\rangle$. Quando la base scelta da Bob per la misura non coincide con quella scelta da Alice per la codifica, ossia quando $b_j \neq b'_j$ per un certo $j = 1, \ldots, 4n$, vengono scartati i corrispondenti bits a_j e a'_j dalle stringhe a e a', poiché c'è il 50% di probabilità che siano diversi. Quando le basi scelte dai due soggetti coincidono, ossia $b_j = b'_j$, i corrispondenti

bits di Alice e Bob, a_j e a'_j, vengono invece conservati, in quanto, al netto di interferenze esterne, deve essere $a_j = a'_j$. Alla fine di questo procedimento, le stringhe a e a' avranno (circa) $2n$ bits ciascuna[7], con $a = a'$ al netto di interferenze esterne come la presenza di Eve.

5: Alice seleziona un sottoinsieme casuale di n bits, che serviranno come controllo della presenza di Eve. Alice e Bob confrontano dunque sul canale classico pubblico il valore degli n bits selezionati: se questi differiscono al di sopra di una certa soglia stabilita in precedenza il protocollo fallisce ed è necessario ripetere tutta la procedura usando canali diversi, meno disturbati o più sicuri. Se il protocollo invece ha successo, i rimanenti n bits fungono da chiave privata per cifrare i messaggi secondo un sistema di crittografia a chiave privata.

Il modo più semplice per attaccare il protocollo è il cosiddetto *Intercept-Resend*: Eve genera una stringa b'' di $4n$ bits casuali per determinare la base, Z o X, con cui misurare i $4n$ qubits[8] dello stato $|\psi\rangle$, non potendo copiarli né discriminarli senza perturbare il sistema. I qubits vengono mandati a Bob dopo la misura. Se la base scelta da Eve coincide con quella scelta da Alice, ossia $b''_j = b_j$, lo stato del qubit non cambia e l'azione di Eve non viene rilevata poiché si avrebbe sempre $a'_j = a_j$ dopo la misura di Bob. Ciò accade col 50% di probabilità. Se invece le basi non coincidono, lo stato del qubit viene perturbato e Bob ha il 50% di probabilità di scoprire l'azione di Eve, a seconda che scelga la sua stessa base o meno. Infatti, se la base di Bob è la stessa di Eve, allora si ha $a'_j = a_j$; se invece la Base di Bob è diversa da quella di Eve, e quindi uguale a quella di Alice ($b''_j \neq b'_j = b_j$), si ha necessariamente $a'_j \neq a_j$. La probabilità totale di rilevare l'azione di Eve è allora:

$$P(a'_j \neq a_j) = \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4} = 25\%$$

In presenza di un intercettatore, degli n bits di controllo condivisi da Alice e Bob, circa $n/4$ sono diversi. Questa discrepanza permette quindi di stabilire la presenza di un intercettatore ed annullare il protocollo.

[7] Siccome le stringhe b e b' vengono generate casualmente, la probabilità che Alice e Bob usino basi diverse è $P = 1/2$.

[8] Eve può scegliere di misurare meno qubits, avendo meno probabilità di essere scoperta ma anche meno probabilità di decifrare il messaggio avendo meno bits per la chiave.

Conclusione

La meccanica quantistica è una delle teorie scientifiche più eleganti e più di successo in possesso dell'umanità. Dalla sua nascita e con la sua maturazione, fin dai primi decenni del XX secolo, è divenuta il framework di riferimento per svariati campi della fisica e della scienza in generale. Fra le altre cose, ha permesso una conoscenza più profonda delle forze fondamentali della natura, con l'elaborazione del Modello Standard, ed ha rivoluzionato la comprensione di molti fenomeni legati alla fisica della materia, dall'ottica alla superconduttività. L'ispirazione che ha mosso il presente lavoro di tesi è stata quella di indagare i contributi della meccanica quantistica nella teoria dell'informazione e nella computazione. In particolare, esplorare come la natura quantistica della materia potesse essere sfruttata per risolvere una certa categoria di problemi. Si è visto come la computazione quantistica sia in grado di risolvere in maniera efficiente alcuni problemi **NP**-complessi, come eseguire una trasformata di Fourier quantistica, ottenendo anche un vantaggio esponenziale rispetto ad una macchina classica. Individuare e studiare delle classi di complessità quantistiche risulta quindi una prospettiva interessante, per comprendere le potenzialità e l'applicabilità della computazione quantistica rispetto a quella classica. La classe di complessità dei problemi risolvibili in maniera efficiente da un computer quantistico, ammettendo un grado di probabilità limitato, è detta **BQP**. Per ora ciò che è noto è che $\mathbf{BQP} \supset \mathbf{P}$, ma non si sa invece se $\mathbf{BQP} \supseteq \mathbf{NP}$, ossia se un computer quantistico è in grado di risolvere in maniera efficiente qualsiasi problema in **NP**. Nel caso particolare trattato dell'algoritmo di Shor, si è visto come, riducendo il problema della fattorizzazione all'order-finding, e, in ultima analisi, all'eseguire una trasformata di Fourier quantistica, si riesce ad ottenere una soluzione polinomialmente efficiente. Si è visto poi come questo possa costituire potenzialmente una minaccia per gli attuali sistemi di sicurezza informatica basati sulla crittografia RSA. Ciò, ovviamente, disponendo di hardware quantistici con un numero sufficiente di qubits, ad oggi non disponibili. Una soluzione a questo problema può consistere nell'utilizzo di problemi diversi dalla fattorizzazione, di cui non è ancora nota una soluzione efficiente tramite algoritmo quantistico. Una soluzione alternativa è invece offerta dalla meccanica quantistica stessa, attraverso dei protocolli di scambio quantistico della chiave, dimostratamente sicuri.

Capitolo 4

Appendice

4.1 Complementi di Teoria dei Numeri

In questa sezione si dimostrano i risultati di teoria dei numeri utilizzati nella sezione sull'algoritmo RSA.

Teorema cinese del resto. Siano $m_1, \ldots, m_n \in \mathbb{N}$ numeri interi positivi fra loro coprimi, ossia tali per cui $\gcd(m_i, m_j) = 1,\ \forall i \neq j$. Allora il sistema di equazioni

$$\begin{cases} x = a_1 \pmod{m_1} \\ x = a_2 \pmod{m_2} \\ \vdots \\ x = a_n \pmod{m_n} \end{cases}$$

ha una soluzione. Inoltre, qualsiasi coppia di soluzioni (x e x') è equivalente modulo $M = m_1 m_2 \ldots m_n$, ossia $x = x' \pmod{M}$.

Dimostrazione
La prima parte della dimostrazione consiste nel costruire una soluzione del sistema di equazioni in oggetto. Si definisce quindi $M_i \equiv M/m_i$, e si osserva che M_i ed m_i sono coprimi. Si può quindi definire l'inverso di M_i modulo m_i:

$$N_i \equiv M_i^{-1} \pmod{m_i} \implies N_i M_i = 1 \pmod{m_i}$$

Inoltre M_i è per definizione divisibile per $m_j, \forall j \neq i$, da cui si ricava che

$$N_i M_i = 0 \pmod{m_j}\ \forall j \neq i\ .$$

Si definisce $x \equiv \sum_{i=1}^{n} a_i N_i M_i$ e si mostra che è effettivamente una soluzione del sistema. Utilizzando le equazioni precedentemente ricavate, e passando

all'aritmetica modulo m_j si trova che

$$x = \sum_{i=1}^{n} a_i N_i M_i \pmod{m_j} = \sum_{i=1}^{n} a_i \delta_{ij} \pmod{m_j} = a_j \pmod{m_i} .$$

Si suppone ora che x e x' siano due soluzioni del sistema di equazioni. Dalla relazione precedente ne segue che $x - x' = 0 \pmod{m_i}$ e quindi m_i divide $x - x' \quad \forall i = 1, \cdots, n$. Siccome poi gli m_i sono fra loro coprimi, anche il prodotto $M = m_1 \dots m_n$ divide $x - x'$. Da ciò segue infine

$$x - x' = 0 \pmod{M} \implies x = x' \pmod{M} .$$

Piccolo teorema di Fermat. Sia p un numero primo e $a \in \mathbb{N}$ un qualsiasi numero intero. Allora $a^p = a \pmod{p}$. Inoltre, se a non è divisibile per p si ha $a^{p-1} = 1 \pmod{p}$.

Dimostrazione
La prima parte del teorema si dimostra per induzione su a. Per $a = 1$ si ottiene l'uguaglianza $a^p = 1 = a \pmod{p} = 1 \pmod{p}$, come richiesto. Si considera vera l'ipotesi induttiva, ossia $a^p = a \pmod{p}$, e si considera il caso $a + 1$. Usando l'espansione binomiale si ha

$$(1 + a)^p = \sum_{k=0}^{p} \binom{p}{k} a^k . \tag{4.1.1}$$

Dalla definizione di coefficiente binomiale si ottiene

$$\binom{p}{k} = \frac{p!}{k!(p-k)!} \implies p(p-1)\cdots(p-k+1) = \binom{p}{k} k(k-1)\cdots 1$$

Per $k \geq 1$, il membro sinistro dell'equazione (e di conseguenza il destro) è divisibile per p. Inoltre, per $k \leq p - 1$, il termine $k(k-1)\cdots 1$ del membro destro non è divisibile per p. Ne segue che, per $1 \leq k \leq p - 1$, il coefficiente binomiale $\binom{p}{k}$ è divisibile per p. Dunque, applicando l'aritmetica modulo p, tutti i termini della somma nell'equazione (4.1.1) si cancellano, tranne quelli per $k = 0$ e per $k = p$, ossia il primo e l'ultimo. Si ottiene allora $(1 + a)^p = (1 + a^p) \pmod{p}$. Applicando l'ipotesi induttiva, ossia $a^p = a \pmod{p}$, si ottiene $(1+a)^p = (1+a) \pmod{p}$, che conclude il procedimento induttivo. La seconda parte del teorema segue immediatamente dalla prima. Infatti, se a non è divisibile per p, allora se ne può definire l'inverso modulo p, da cui si ottiene $a^{p-1} = a^{-1}a^p = a^{-1}a \pmod{p} = 1 \pmod{p}$.

Piccolo teorema di fermat generalizzato. Sia $n \in \mathbb{N}$ un intero coprimo di a. Allora vale $a^{\phi(n)} = 1 \pmod{n}$, dove $\phi(n)$ è la funzione toziente di Eulero di n. Tale generalizzazione del piccolo teorema di Fermat è dovuta a Eulero.

Dimostrazione
Sia n un generico numero naturale che si scrive nella sua forma fattorizzata $n = p_1^{\alpha_1} \cdots p_m^{\alpha_m}$. Si ricorda che $\phi(n)$ rappresenta la cardinalità dell'insieme $R(n) = \{r \in \mathbb{N} | 0 < r < n, \gcd(r, n) = 1\}$ dei numeri interi coprimi e minori di n. Dato un numero primo p, si può agilmente dimostrare la validità delle seguenti relazioni: $\phi(p) = p - 1$, $\phi(p^\alpha) = p^{\alpha-1}(p-1)$, $\phi(n) = \prod_i \phi(p_i^{\alpha_i})$. Per prima cosa si dimostra per induzione su α che $a^{\phi(p^\alpha)} = 1 \pmod{p^\alpha}$. Per $\alpha = 1$ si ottiene $a^{\phi(p)} = a^{p-1} = 1 \pmod{p}$, che è la tesi già dimostrata del piccolo teorema di Fermat. Assumendo vera l'ipotesi induttiva per $\alpha \geq 1$ si ha

$$a^{\phi(p^\alpha)} = 1 + kp^\alpha, k \in \mathbb{N} \ .$$

Dunque, per il caso $\alpha + 1$ si ottiene:

$$a^{\phi(p^{\alpha+1})} = a^{p^\alpha(p-1)} = a^{p\phi(p^\alpha)} = (a^{\phi(p^\alpha)})^p = (1 + kp^\alpha)^p = 1 + \sum_{j=1}^{p} \binom{p}{j} k^j p^{j\alpha},$$

dove nell'ultimo passaggio si è utilizzata l'espansione binomiale esplicitando il caso per $j = 0$. Riprendendo il procedimento fatto nella dimostrazione del precedente teorema, si può dimostrare come $p^{\alpha+1}$ divide ogni termine della somma nella precedente espressione (p divide $\binom{p}{j}$ e p^α divide $p^{j\alpha}$). Passando all'aritmetica modulo $p^{\alpha+1}$ si ottiene allora

$$a^{\phi(p^{\alpha+1})} = 1 \pmod{p^{\alpha+1}} \ ,$$

e si conclude il procedimento induttivo. Per ottenere la tesi del teorema è sufficiente osservare che $\phi(n) = \prod_i \phi(p_i^{\alpha_i})$ è un multiplo di $\phi(p_i^{\alpha_i})$ $\forall i = 1, \ldots, m$, e quindi che vale la relazione $a^{\phi(n)} = 1 \pmod{p_i^{\alpha_i}}$ $\forall i = 1, \ldots, m$. Applicando il teorema cinese del resto si ottiene infine

$$a^{\phi(n)} = 1 \pmod{p_1^{\alpha_1} \cdots p_m^{\alpha_m}} = 1 \pmod{n} \ .$$

Lemma (rappresentazione del massimo comun divisore). Il massimo comun divisore di due interi a e b è il più piccolo intero positivo che si può scrivere nella forma $\gcd(a, b) = ax + by$, dove x e y sono due numeri interi non necessariamente positivi. Sia inoltre c un intero che divide sia a che b. Allora c divide $\gcd(a, b)$.

Dimostrazione
Sia $s = ax + by$ il più piccolo intero positivo scrivibile come combinazione lineare di a e b. Siccome per definizione $\gcd(a, b)$ è un divisore sia di a che di b, segue che $\gcd(a, b)$ è un divisore anche di s. Dunque $\gcd(a, b) \leq s$. Per completare la dimostrazione si può dimostrare che $\gcd(a, b) \geq s$, mostrando che s è un divisore sia di a che di b. Si procede per assurdo. Si suppone per assurdo che s non è un divisore di a. Esiste dunque un resto r, con $1 \leq r \leq s-1$, tale per cui si ha $a = ks + r$. Sapendo che $s = ax + by$ si ottiene dalla precedente espressione $r = a - ks = a(1 - kx) + b(-ky)$, ossia un numero intero positivo minore di s $(1 \leq r \leq s - 1)$ scrivibile come combinazione lineare di a e b. Questo contraddice l'ipotesi fatta per s. Dunque, s deve essere un divisore di a, e per simmetria di b, da cui $\gcd(a, b) \geq s \implies \gcd(a, b) = s$. Siccome poi c divide sia a che b, segue che c divide anche $ax + by = \gcd(a, b)$. Questo conclude la dimostrazione.

Teorema (Algoritmo di Euclide). Siano a e b due interi, e sia r il resto della divisione fra a e b, tale che $a = kb + r$. Allora, supposto che $r \neq 0$, $\gcd(a, b) = \gcd(b, r)$.

Dimostrazione
Si dimostra l'uguaglianza mostrando che ciascun membro divide l'altro. Siccome $\gcd(a, b)$ per definizione divide a e b, ed $r = a - kb$, risulta che $\gcd(a, b)$ divide r. Per il lemma precedente $\gcd(a, b)$ divide $\gcd(b, r) \implies \gcd(a, b) \leq \gcd(b, r)$. Allo stesso modo, dato che $a = kb + r$, e $\gcd(b, r)$ divide b ed r per definizione, segue che $\gcd(b, r)$ divide a. Per il lemma precedente allora $\gcd(b, r)$ divide $\gcd(a, b) \implies \gcd(b, r) \leq \gcd(a, b)$.

L'algoritmo è un'immediata conseguenza di questo risultato. Dati due interi $a > b$, con $a = kb + r$, per il teorema precedente si ha $\gcd(a, b) = \gcd(b, r)$. Si ripete il procedimento per la divisione fra b e r: $b = k_1 r + r_1 \implies \gcd(a, b) = \gcd(b, r) = \gcd(r, r_1)$. Si itera questo procedimento finché non si ottiene un resto nullo. Dopo n passaggi si ha: $r_{n-1} = k_n r_n \implies \gcd(a, b) = \ldots = \gcd(r_{n-1}, r_n) = \gcd(r_n, 0) = r_n$. L'algoritmo fornisce dunque il risultato r_n come massimo comun divisione di a e b.

4.2 Osservazioni sull'Order-finding

Facendo riferimento a quanto detto nella sezione 2.2 sull'order-finding, si svolge ora lo stesso procedimento in un altro modo, facendo agire direttamente il gate $U_{m,N}$ sullo stato $|1\rangle = |0\rangle^{L-1} \otimes |1\rangle$ della base computazionale

di $\mathbb{C}^{2^L}$:

$$\frac{1}{2^{t/2}} \sum_{x=0}^{2^t-1} |x\rangle\, U_{m,N}^x\, |1\rangle = \frac{1}{2^{t/2}} \sum_{x=0}^{2^t-1} |x\rangle\, |m^x \pmod N\rangle\,. \tag{4.2.1}$$

Applicando la Trasformata di Fourier quantistica inversa al primo registro, e ponendo $Q = 2^t$, si ottiene:

$$\overset{QFT^\dagger}{\to} \frac{1}{Q} \sum_{x=0}^{Q-1} \sum_{y=0}^{Q-1} e^{-2\pi i x y/Q}\, |y\rangle\, |m^x \pmod N\rangle\,. \tag{4.2.2}$$

Definendo poi

$$|\Gamma(y)\rangle = \sum_{x=0}^{Q-1} e^{-2\pi i x y/Q}\, |m^x \pmod N\rangle \tag{4.2.3}$$

si giunge all'espressione:

$$|Reg1\rangle\, |Reg2\rangle = \frac{1}{Q} \sum_{y=0}^{Q-1} |y\rangle\, |\Gamma(y)\rangle = \frac{1}{Q} \sum_{y=0}^{Q-1} \|\Gamma(y)\|\, |y\rangle\, \frac{|\Gamma(y)\rangle}{\|\Gamma(y)\|} \quad . \tag{4.2.4}$$

Misurando il primo registro si ottiene come stato finale del processo:

$$|y\rangle\, \frac{|\Gamma(y)\rangle}{\|\Gamma(y)\|}, \qquad Prob(y) = \left(\frac{\|\Gamma(y)\|}{Q}\right)^2 \tag{4.2.5}$$

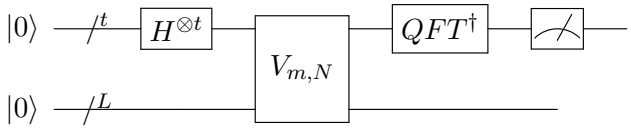
per un certo valore $0 \le y \le Q-1$ e con probabilità $Prob(y)$. Per quanto visto precedentemente, dal valore $y \approx \frac{s}{r} \cdot 2^t$, ottenuto dalla lettura del primo registro, si può estrarre l'ordine r calcolando la frazione minima più vicina a y/Q. Si dimostra infatti [5] che anche in questo caso vale $|s/r - y/Q| \le 1/(2Q) \le 1/(2N^2) << 1$ per N sufficientemente grande. Dunque y/Q è una buona stima di s/r per un certo s, con $0 \le s \le r-1$. Si può poi dimostrare che la probabilità di ottenere un outcome y dalla misura del primo registro è data da [14, 3]:

$$Prob(y) = \left(\frac{\|\Gamma(y)\|}{Q}\right)^2 \approx \frac{r \sin^2(\pi y)}{Q^2 \sin^2(\pi y r/Q)} \quad ,$$

dove $Q = 2^t$. Siccome y è un numero intero, $\sin^2(\pi y) = 0$ e $Prob(y) \neq 0 \iff yr = 0 \pmod Q \implies yr = kQ$, ossia Q è un multiplo di ry, con $k \in \mathbb{N}$. Prendendo il limite per $y \to kQ/r$, si ottiene quindi:

$$\implies Prob(y) \approx \begin{cases} 0 & ry \neq 0 \pmod Q \\ \frac{1}{r} & ry = 0 \pmod Q \end{cases} \tag{4.2.6}$$

Questo risultato è perfettamente coerente con quanto ottenuto nella relazione (2.2.7), come ci si aspetta. Si può inoltre notare che lo stato dell'equazione (2.2.4), e quindi tutto il procedimento successivo, può essere ottenuto inizializzando il secondo registro nello stato $|0\rangle$ e utilizzando il gate $V_{m,N}|x\rangle|k\rangle = |k \oplus m^x \pmod N\rangle$, come visto nel paragrafo 1.4:



Infatti:

$$|0\rangle|0\rangle \xrightarrow{H} \frac{1}{2^{t/2}}\sum_{x=0}^{2^t-1}|x\rangle|0\rangle \xrightarrow{V_{m,N}} \frac{1}{2^{t/2}}\sum_{x=0}^{2^t-1}V_{m,N}|x\rangle|0\rangle = \tag{4.2.7}$$

$$= \frac{1}{2^{t/2}}\sum_{x=0}^{2^t-1}|x\rangle|0 \oplus m^x \pmod N\rangle\,. \tag{4.2.8}$$

Capitolo 5

Bibliografia

[1] Michael A. Nielsen and Isaac L. Chuang. Quantum Computation and Quantum Information. *Cambridge University Press*, 10th Anniversary Edition.

[2] Fabio Benatti, Mark Fannes, Roberto Floreanini, Dimitri Petritis. Quantum Information, Computation and Cryptography. *Springer*

[3] Mikio Nakahara, Tetsuo Ohmi. Quantum Computing. From Linear Algebra to Physical realizations. *CRC Press, 2008*

[4] Sito della libreria Qiskit per Phyton. `qiskit.org/learn`

[5] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. *Proceedings 35th annual symposium on foundation of computer science*, 124-134, 1994

[6] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review* 41 (2), 303-332, 1999

[7] Arvid J Bessen. Lower bound for quantum phase estimation. *Physical Review* A 71 (4), 2005

[8] G. H. Hardy, E. M. Wright. An Introduction to the Theory of Numbers. *Oxford Science Publication*, 5th edition. Cambridge University Press.

[9] Arjen K. Lenstra e H. W. Lenstra. The development of the number field sieve. *Lecture Notes in Math*, 1993. Springer-Verlag.

[10] C. H. Bennet and G. Brassard. Quantum cryptography: Public key distribution and coin tossing. In *Proceedings of IEEE International conference on Computers, Systems and Signal Processing*, 175-179. New York, 1984.

[11] C. E. Shannon. Communication theory of secrecy systems. *The Bell system technical journal*, 1949.

[12] R. L. Rivest, A. Shamir, L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 1978.

[13] R. Cleve, A. Ekert, C. Macchiavello and M. Mosca. Quantum Algorithms Revisited. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, vol. 454, n. 1969*, 8 January 1998.

[14] Michael R. Geller, Zhongyuan Zhou. Factoring 51 and 85 with 8 qubits. *Department of Physics and Astronomy, University of Georgia,* 2013.

[15] T. Monz, D. Nigg, E. A. Martinez, M. F. Brandl, P. Schindler, R. Rines, S. X. Wang, I. L. Chuang, and R. Blatt. Realization of a scalable Shor algorithm. *Science 351 (6277), 1068-1070,* 2016.

[16] Richard P. Feynman. Simulating physics with computers, 1981. *International Journal of Theoretical Physics, 21(6/7).*