ALMA MATER STUDIORUM – UNIVERSITÀ DI BOLOGNA

Corso di Laurea in Ingegneria e Scienze Informatiche

Post-Quantum Cryptography

Tesi di laurea in:
Crittografia

Relatore
Prof. Margara Luciano

Candidato
Olivieri Michele

III Sessione di Laurea
Anno Accademico 2024-2025

Abstract

La sicurezza digiatle moderna pone le sue fondamenta sulla crittografia a chiave pubblica, che a sua volta delega la sua robustezza alla difficoltà computazione di problemi matematici come il problema della fattorizzazione e del logaritmo discreto. Tuttavia, l'avvento dei computer quantistici minaccia di compromettere la sicurezza di questi sistemi, poiché algoritmi quantistici come quello di Shor possono risolvere tali problemi in tempo polinomiale sfruttando la trasformata di Fourier quantistica per individuare la periodicità di funzioni modulari. Sebbe i comuper quantistici attuali non siano ancora in grado di violare chiavi crittografiche reali, il modello di minaccia "harvest now, decrypt later" rende la transizione urgente già oggi.

In risposta, la crittografia post-quantistica sviluppa algoritmmi fondati su famiglie di problemi matematici considerati resistenti anche ad attacchi quantistici: reticoli algebrici, codici correttori di errori e funzioni hash. Il NIST ha pubblicato nel 2024 i primi standar ufficiali ML-KEM, ML-DSA e SLH-DSA, segnando l'inizio concreto della transizione. Questa tesi analizza i fondamenti teorici di questa evoluzione, dall'impatto di Shor sulla crittografia classica alle soluzioni post-quantistiche, evidenziando come la migrazione verso questi nuovi paradigmi rappresenti una delle sfide più attuali e urgenti per la sicurezza informatica globale del prossimo decennio.

The enemy knows the system.
Claude E. Shannon

Contents

List of Figures

Chapter 1

Introduction

Contesto e motivazioni Per comprendere l'importanza della crittografia post-quantistica, è fondamentale analizzare il contesto in cui essa si inserisce e le motivazioni che ne hanno guidato lo sviluppo.

La crittografia classica, come visto a lezione, pone le sue fondamenta su problemi computazionalmente difficili per i quali non esistono algoritmi efficienti in grado di risolverli in tempi polinomiali.

Tuttavia, l'emergere dei computer quantistici ha introdotto una nuova dimensione nel panorama della sicurezza informatica. Questi dispositivi sfruttando i principi della meccanica quantistica sono in grado di eseguire calcoli in modo radicalmente diverso rispetto ai computer classici, rendendoli potenzialmente capaci di risolvere in tempi polinomiali quei problemi matematici che risultano intrattabili per le macchine convenzionali [NC10].

Minaccia dei computer quantistici: Perché è necessaria? L'ipotesi di una minaccia quantistica emerge già nel ventesimo secolo, quando nel 1994 l'algoritmo di Shor [Sho94] dimostra che un computer quantistico sufficientemente potente potrebbe fattorizzare grandi numeri interi e calcolare logaritmi discreti in tempo polinomiale. Questo rende vulnerabili algoritmi come RSA, ECC e DSA, che costituiscono la base della sicurezza informatica moderna.

Per molti anni il problema è rimasto solo teorico, perché il calcolo quantistico non aveva applicazioni pratiche. Lo scenario è cambiato con i recenti progressi tecnologici e lo sviluppo dei primi prototipi avanzati da parte di leader industriali

come Google e Microsoft, che hanno riportato risultati significativi con i loro progetti: Willow[1] di Google e il processore Maiorana 1[2] di Microsoft. Questi progetti hanno quindi portato alla luce dei veri calcolatori quantistici funzionanti e nonostante il limitato numero di qbit stabili sia insufficiente per applicazioni pratiche su larga scala, questo segna una svolta nel settore.

Sebbene le macchine quantistiche siano ancora in fase sperimentale, diversi scienziati ritengono che la loro costruzione su larga scala sia ormai una sfida principalmente ingegneristica, e alcuni prevedono la loro maturazione entro i prossimi vent'anni.

Considerando che l'attuale infrastruttura crittografica ha richiesto quasi due decenni per essere implementata, risulta necessario iniziare da subito la transizione verso sistemi progettati per resistere al calcolo quantistico [Ber09].

Obiettivi della crittografia post-quantistica L'obiettivo della crittografia post-quantistica è quindi sviluppare algoritmi crittografici sicuri sia contro i computer quantistici che classici, garantendo così un futuro sicuro anche in un'era dominata dai computer quantistici.

La crittografia post-quantistica non si limita a sostituire gli algoritmi vulnerabili, ma mira a costruire un'infrastruttura di sicurezza robusta e duratura, capace di adattarsi alle sfide tecnologiche future mantenendo la compatibilità con i protocolli e le reti esistenti[3].

Structure of the Thesis Lo scopo della relazione invece è quindi quello di fornire una panoramica completa, introducendo in primo luogo i fondamenti della crittografia classica, per capire dove risiedono le vulnerabilità che i computer quantistici sono in grado di sfruttare. Una volta capiti tali principi, procederemo ad analizzare Shor, il suo impatto su tali algoritmi ed infine le soluzioni proposte dalla crittografia post-quantistica.

[1] Google Quantum AI, "Willow: A quantum computing milestone", Dicembre 2024. Disponibile su: blog.google

[2] Microsoft Quantum, "Maiorana 1: The first milestone on the path to a quantum supercomputer", Ottobre 2024. Disponibile su: www.microsoft.com

[3] NIST, "Post-Quantum Cryptography Standardization", `https://csrc.nist.gov/projects/post-quantum-cryptography`

Chapter 2

Fondamenti di crittografia classica

Come anticipato, per comprendere le sfide poste dai computer quantistici alla crittografia moderna, è fondamentale conoscerne i principi di base dietro i principali algoritmi crittografici attualmente in uso, in particolar modo ci concentreremo sui protocolli a chiave pubblica.

2.1 Crittografia a chiave pubblica: RSA, ECC

La crittografia a chiave pubblica, introdotta nel 1976 da Diffie, Hellman e Merkle [DH76], costituisce una svolta fondamentale nel panorama della sicurezza informatica moderna. A differenza dei sistemi simmetrici, che vincolano mittente e destinatario alla condivisione di un unico segreto, i protocolli asimmetrici impiegano una coppia di chiavi: una pubblica k_{pub}, liberamente distribuibile, e una privata k_{prv}, mantenuta segreta dal proprietario. La sicurezza di questo sistema si basa sulla difficoltà di risalire alla chiave privata a partire da quella pubblica. Le funzioni di cifratura C e decifratura D sono note a tutti, e per ogni messaggio m deve valere:

$$D(C(m; k_{\text{pub}}); k_{\text{prv}}) = m.$$

Il funzionamento di questo sistema si basa sulle funzioni one-way trapdoor: operazioni matematiche semplici da eseguire in una direzione, ma computazionalmente intrattabili da invertire senza la conoscenza di una informazione specifica (la "trappola").

La teoria dei numeri e l'algebra modulare forniscono il substrato matematico necessario per generare tali funzioni; a seconda del problema matematico sottostante, si distinguono i vari algoritmi di crittografia asimmetrica oggi in uso.

Richiami di algebra modulare L'aritmetica modulare è un sistema in cui i numeri si riavvolgono entro un intervallo fissato da un modulo n. Quando un valore supera (o scende sotto) questo intervallo, viene riportato all'interno prendendo il resto della divisione per n.

Un esempio quotidiano è l'orologio: in un sistema a 12 ore il modulo è 12. Se sono le 10 e aggiungo 4 ore, il risultato non è 14, ma 2, perché:

$$14 \equiv 2 \pmod{12}.$$

Per calcolare $c = a \bmod b$ si considera il resto della divisione intera tra a e b, ottenendo un valore sempre compreso tra 0 e $b - 1$, esempio: $6 \bmod 4 = 2$.

Problemi difficili

- **Fattorizzazione:** dati p, q è facile calcolare $n = pq$; dato n è difficile trovare p e q.
- **Radice modulare:** dato $y = x^z \bmod s$ invertire la potenza è difficile senza conoscere $\varphi(s)$.
- **Logaritmo discreto:** data $y = x^z \bmod s$ trovare z è computazionalmente difficile.

2.1.1 RSA

Il cifrario RSA, proposto da Rivest, Shamir e Adleman nel 1978 [RSA78], è il sistema crittografico a chiave pubblica più diffuso e studiato. La sua sicurezza si fonda sulla difficoltà computazionale della fattorizzazione di numeri interi molto grandi[1].

[1] Materiale didattico del corso di Crittografia, Università di Bologna, a.a. 2024/2025

Generazione delle chiavi Ogni utente genera la propria coppia di chiavi attraverso i seguenti passaggi:

1. Scelta di due numeri primi p e q molto grandi
2. Calcolo di $n = pq$ e della funzione di Eulero $\phi(n) = (p-1)(q-1)$
3. Scelta di un intero e minore di $\phi(n)$ e coprimo con esso
4. Calcolo dell'intero d, inverso moltiplicativo di e modulo $\phi(n)$

La chiave pubblica è la coppia (e, n), mentre la chiave privata è d. La cifratura di un messaggio m avviene calcolando $c = m^e \bmod n$, mentre la decifratura richiede il calcolo di $m = c^d \bmod n$.

La correttezza dell'algoritmo è garantita dal teorema di Eulero: poiché $ed \equiv 1 \pmod{\phi(n)}$, si ha $ed = 1 + k\phi(n)$ per qualche intero k, e quindi:

$$m^{ed} \bmod n = m^{1+k\phi(n)} \bmod n = m \cdot (m^{\phi(n)})^k \bmod n = m \bmod n$$

Sicurezza e dimensioni delle chiavi La sicurezza di RSA dipende da l'impossibilità pratica di fattorizzare n quando questo è sufficientemente grande. Conoscendo la fattorizzazione $n = pq$, un attaccante potrebbe infatti calcolare $\phi(n)$ e di conseguenza la chiave privata d.

Attualmente, le dimensioni delle chiavi considerate sicure sono di almeno 2048 bit, con raccomandazioni crescenti verso 4096 bit per applicazioni che richiedono sicurezza a lungo termine [Nat20]. Chiavi di 1024 bit sono considerate obsolete e vulnerabili ad attacchi con risorse computazionali moderne.

2.1.2 Crittografia su Curve Ellittiche (ECC)

La crittografia su curve ellittiche, sviluppata indipendentemente da Neal Koblitz e Victor Miller nel 1985 [Kob87, Mil85], offre un'alternativa matematicamente elegante e computazionalmente efficiente a RSA.

Fondamenti matematici Una curva ellittica su un campo finito $\mathbb{Z}_p$ (con p primo e $p > 3$) è definita dall'equazione di Weierstrass in forma normale:

$$y^2 = x^3 + ax + b$$

dove $a, b \in \mathbb{Z}_p$ soddisfano la condizione $4a^3 + 27b^2 \bmod p \neq 0$, che garantisce l'assenza di punti singolari sulla curva[2].

L'insieme dei punti (x, y) che soddisfano questa equazione, insieme al punto all'infinito $\mathcal{O}$, forma un gruppo abeliano additivo. È possibile definire un'operazione di addizione tra punti della curva tale che, dati due punti P e Q, la loro somma $P + Q$ sia ancora un punto della curva.

Per punti distinti $P = (x_P, y_P)$ e $Q = (x_Q, y_Q)$, con $P \neq -Q$, si ha:

$$\lambda = \frac{y_Q - y_P}{x_Q - x_P}, \quad x_S = \lambda^2 - x_P - x_Q, \quad y_S = -y_P + \lambda(x_P - x_S)$$

dove $S = P + Q = (x_S, y_S)$. Nel caso di raddoppio di un punto ($P = Q$), il coefficiente angolare diventa $\lambda = \frac{3x_P^2 + a}{2y_P}$.

Il problema del logaritmo discreto La sicurezza di ECC si basa sulla difficoltà del problema del logaritmo discreto su curve ellittiche (ECDLP): dati due punti P e Q sulla curva, trovare l'intero k tale che $Q = kP$, dove kP denota l'addizione di P con se stesso k volte.

La moltiplicazione scalare $Q = kP$ è computazionalmente efficiente (tempo polinomiale), mentre il problema inverso è considerato intrattabile: tutti gli algoritmi classici noti hanno complessità esponenziale nella dimensione della chiave.

[2]Materiale didattico del corso di Crittografia, Università di Bologna, a.a. 2024/2025

Vantaggi rispetto a RSA ECC offre lo stesso livello di sicurezza di RSA con chiavi significativamente più corte. Una chiave ECC di 256 bit fornisce una sicurezza paragonabile a una chiave RSA di 3072 bit [Nat20]. Questo si traduce in:

- Minore occupazione di memoria e larghezza di banda
- Operazioni crittografiche più veloci
- Minore consumo energetico, cruciale per dispositivi mobili e IoT

2.1.3 Sicurezza classica

Entrambi gli algoritmi sono considerati sicuri nell'ambito del calcolo classico per dimensioni di chiave appropriate. La loro robustezza deriva dalla complessità computazionale dei problemi matematici sottostanti:

- **Fattorizzazione per RSA**: il miglior algoritmo classico noto è il General Number Field Sieve (GNFS), con complessità sub-esponenziale [LL93]
- **ECDLP per ECC**: gli algoritmi più efficienti, come il metodo rho di Pollard, hanno complessità completamente esponenziale $O(\sqrt{n})$, dove n è l'ordine del gruppo [Pol78]

Questa differenza nella complessità degli attacchi spiega perché ECC richiede chiavi più corte per garantire lo stesso livello di sicurezza.

RSA e ECC costituiscono oggi la base dell'infrastruttura di sicurezza digitale globale, utilizzati in TLS/SSL per la sicurezza web, in SSH per l'accesso remoto sicuro, nella firma digitale di documenti e software, e in numerose altre applicazioni critiche.

2.2 Limiti rispetto ai computer quantistici

Avendo introdotto i fondamenti della crittografia classica possiamo provare ora ad analizzare le vulnerabilità di questi algoritmi in particolar modo rispetto al calcolo quantistico. Infatti nei prossimi capitoli capiremo quali sono i vantaggi che i computer quantistici offrono rispetto a quelli classici e come questi possono compromettere la sicurezza degli algoritmi classici. in particolare esamineremo l'algoritmo di Shor e il suo impatto su RSA.

Chapter 3

Computazione quantistica e impatto sulla crittografia classica

Entriamo ora nel vivo della relazione, sviscerare i segreti dietro l'algoritmo di Shor per capire in che modo rompere il protocllo RSA appena descritto. Per farlo avremo prima bisogno di introdurre alcuni concetti di base della computazione quantistica che shor utilizza per ottenere i suoi risultati. Una volta compresi questi concetti potremo procedere alla spiegazione dell'algoritmo vero e proprio.

3.1 Nozioni base di computazione quantistica

La computazione quantistica rappresenta un paradigma di calcolo che sfrutta i principi della meccanica quantistica per elaborare l'informazione in modi che non sono possibili con i computer classici[NC10, Sezione 1.1]. Di questi principi di meccanica quantistica non esiste un corrispettivo diretto nel modelli di calcolo classico, e proprio per questo motivo che ne analizziamo gli effetti e le conseguenze.

3.1.1 Richiami essenziali di meccanica quantistica

In questa sezione non si fornisce una trattazione formale della teoria, ma si introducono i concetti essenziali necessari a comprendere il funzionamento dei computer quantistici e dei meccanismi che consentono loro di superare i limiti della computazione classica.

Il primo concetto fondamentale è quello di ***stato quantistico***. Mentre un bit classico può assumere esclusivamente i valori 0 o 1, un sistema quantistico può trovarsi in una *sovrapposizione* di stati[NC10, Sezione 1.2]. Questo non significa che sia 0 e 1 "contemporaneamente" in senso magico ma semplicemente che lo stato di un qubit può essere descritto come una combinazione lineare degli stati base $|0\rangle$ e $|1\rangle$:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$$

con coefficienti complessi che ne determinano le probabilità di osservazione[NC10, p. 13].

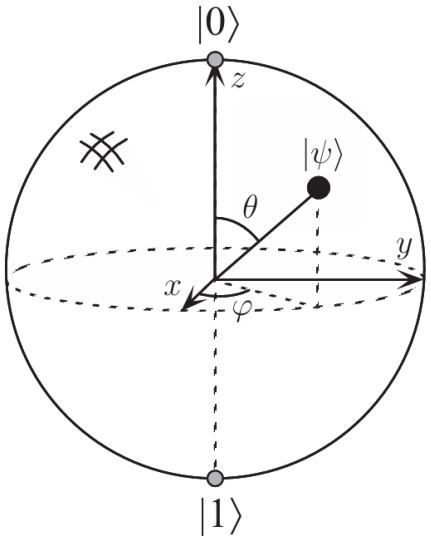

Figure 3.1: Bloch sphere representation of a qubit.

Un secondo concetto chiave è l'***entanglement***, una forma di correlazione quantistica tra più qubit per cui lo stato complessivo del sistema non può essere descritto come il prodotto degli stati dei singoli qubit[NC10, Sezione 2.2.8]. In presenza di entanglement, la misura di un qubit influisce istantaneamente sullo stato degli altri, indipendentemente dalla distanza che li separa[NC10, Sezione 1.2.1]. Questa proprietà consente di rappresentare e manipolare informazioni in modo non separabile, ciò significa che le operazioni su un qubit possono avere effetti immediati sugli altri qubit entangled con esso.

Il terzo elemento fondamentale è la ***misura***. Quando uno stato quantistico

viene misurato, la sovrapposizione collassa e il risultato ottenuto è un valore classico[NC10, Sezione 1.2]. L'esito della misura è probabilistico e dipende dallo stato del sistema immediatamente prima dell'osservazione[NC10, Sezione 2.2.3].

Sovrapposizione, entanglement e misura costituiscono i principi alla base della differenza concettuale tra computazione classica e computazione quantistica, e determinano il modo in cui l'informazione viene elaborata in un sistema quantistico[NC10, Sezione 1.3].

Qubit e operazioni quantistiche

L'unità fondamentale di informazione in un computer quantistico è quindi il qubit[NC10, Sezione 1.2]. Dal punto di vista matematico, un qubit è rappresentato come un vettore unitario in uno spazio di Hilbert bidimensionale, mentre un registro composto da n qubit è descritto in uno spazio di dimensione 2^n.

Le operazioni sui qubit sono realizzate mediante *porte quantistiche*, che corrispondono a trasformazioni lineari e reversibili, descritte da operatori unitari. A differenza delle porte logiche classiche, le porte quantistiche agiscono su stati in sovrapposizione, modificando simultaneamente tutte le componenti dello stato quantistico.

Matematicamente, queste porte sono rappresentate come matrici unitarie, il che conferisce loro due proprietà cruciali: la somma delle probabilità dei possibili stati deve essere sempre pari a 1 e ogni operazione deve essere reversibile (ovvero, ogni porta ha una sua inversa che può annullare l'operazione). Le principali porte quantistiche[NC10, Sezioni 1.3.1, 1.3.2, 1.3.4, 1.3.6, 4.2, 4.5, 5.1]:

- porte a bit singolo come: la porta Hadamard (H), che viene usata per mettere i qubit in superposizioni, la porta Pauli-X (equivalente a una NOT classica), e le porte di Pauli(X, Y, Z) che ruotano il vettore.

- porte a più bit che realizzano operazioni condizionate tra due qubit, fondamentali per creare entanglement.

- porta SWAP che scambia lo stato di due qubit, la porta di Fase Controllata (Rk) utile per la QFT.

- porte specializzate costruite per specifici algoritmi a partire da queste.

L'uso combinato di queste porte consente di elaborare, in un singolo passo computazionale, un insieme di stati che cresce esponenzialmente con il numero di qubit. Tale fenomeno è noto come *parallelismo quantistico* e rappresenta una delle principali fonti di vantaggio rispetto al calcolo classico, pur non traducendosi automaticamente in un'accelerazione per qualsiasi problema[NC10, Sezioni 1.4.2, 4.5.4].

Modello di calcolo quantistico

Il funzionamento di un computer quantistico segue un modello di calcolo specifico, distinto da quello deterministico utilizzato nei computer classici. Un algoritmo quantistico è generalmente articolato in tre fasi principali[NC10, Sezioni 1.3.4, 4.6]:

- preparazione dello stato iniziale;
- applicazione di una sequenza di porte quantistiche;
- misura finale del sistema.

Durante la fase di evoluzione, lo stato quantistico del sistema viene trasformato in modo deterministico secondo le leggi della meccanica quantistica. La natura probabilistica emerge esclusivamente al momento della misura, quando lo stato viene proiettato su uno dei possibili risultati osservabili.

La potenza del modello di calcolo quantistico non risiede nella possibilità di valutare esplicitamente tutte le soluzioni di un problema, bensì nella capacità di sfruttare il fenomeno dell'*interferenza quantistica*. Attraverso opportune sequenze di operazioni, le ampiezze associate alle soluzioni corrette possono essere amplificate, mentre quelle delle soluzioni errate vengono attenuate[NC10, Sezioni 1.4.3, 6.1.3].

3.1.2 Collegamento con la crittografia

I concetti introdotti in questa sezione costituiscono il fondamento teorico per analizzare l'impatto della computazione quantistica sulla crittografia moderna. In particolare, l'uso combinato di sovrapposizione, entanglement e interferenza

consente di affrontare in modo efficiente problemi come la fattorizzazione di interi e il calcolo del logaritmo discreto[NC10, Sezioni 1.4, 5.3].

3.2 Algoritmi quantistici rilevanti: Shor

L'algoritmo di Shor rappresenta uno dei risultati più significativi nel campo della computazione quantistica, non solo per le sue implicazioni sulla crittografia, ma anche per la profondità delle idee matematiche che lo compongono[NC10, Sezione 5.3]. Infatti per comprendere appieno il funzionamento di questo algoritmo, è necessario procedere con un'analisi rigorosa che parta dai fondamenti matematici, prescindendo inizialmente dagli aspetti quantistici.

L'obiettivo quindi di questa sezione è chiarire perché l'algoritmo funziona dal punto di vista matematico, e solo in una fase successiva dell'analisi vedremo come viene accelerato dal computer quantistico.

Il problema della fattorizzazione

Il punto di partenza è il problema della fattorizzazione di interi. Dato un intero composto $N = p \cdot q$, dove p e q sono numeri primi di grandi dimensioni, il problema consiste nel determinare p e q conoscendo solamente N. La difficoltà computazionale della fattorizzazione deriva da due caratteristiche fondamentali: in primo luogo, non è noto alcun algoritmo classico in grado di fattorizzare un numero in tempo polinomiale rispetto alla dimensione dell'input; in secondo luogo, i tentativi diretti di fattorizzazione crescono rapidamente con l'aumentare della dimensione di N, rendendo il problema intrattabile per valori sufficientemente grandi.

L'intuizione fondamentale di Shor consiste nel trasformare il problema della fattorizzazione in un problema di natura diversa, caratterizzato da una struttura matematica più favorevole[NC10, Sezione 5.3.2]. Questa trasformazione permette di ricondurre la ricerca dei fattori primi a un problema di teoria dei numeri che, come vedremo, può essere risolto in modo efficiente sfruttando le peculiarità della computazione quantistica.

3.2.1 Dalla fattorizzazione alla teoria dei numeri

La strategia di Shor si basa sull'osservazione che la fattorizzazione può essere ricondotta al problema della determinazione del periodo di una funzione. Questa connessione non è immediata e richiede una serie di passaggi matematici che è necessario esplicitare con precisione.

Il primo passo consiste nella scelta di un numero intero a tale che $1 < a < N$ e $\gcd(a, N) = 1$. La condizione sulla coprimalità è essenziale: se infatti $\gcd(a, N) \neq 1$, avremmo già trovato un fattore non banale di N semplicemente calcolando il massimo comun divisore. Una volta scelto a, si considera la funzione esponenziale modulare definita come:

$$f(x) = a^x \bmod N$$

Questa funzione possiede una proprietà fondamentale: è periodica. Più precisamente, esiste un minimo intero $r > 0$ tale che:

$$a^r \equiv 1 \pmod{N}$$

Questo valore r è chiamato ordine di a modulo N. L'esistenza di tale periodo è garantita dal teorema di Eulero, secondo cui $a^{\phi(N)} \equiv 1 \pmod{N}$, dove $\phi(N)$ è la funzione di Eulero[NC10, Sezione A4.2]. Il periodo cercato è dunque un divisore di $\phi(N)$.

3.2.2 L'utilità del periodo per la fattorizzazione

A questo punto sorge naturale una domanda: perché la conoscenza del periodo r dovrebbe aiutarci a fattorizzare N? La risposta risiede in una proprietà algebrica profonda che collega l'ordine di un elemento alla struttura moltiplicativa del gruppo $(\mathbb{Z}/N\mathbb{Z})^*$[NC10, Sezione 5.3.2].

Se il periodo r soddisfa due condizioni specifiche, ovvero se r è pari e se $a^{r/2} \not\equiv -1 \pmod{N}$, allora è possibile estrarre fattori non banali di N calcolando:

$$\gcd(a^{r/2} - 1, N) \quad \text{e} \quad \gcd(a^{r/2} + 1, N)$$

Questa è la chiave matematica dell'intero algoritmo[IBM24]. Per comprendere

perché questa procedura funziona, è necessario esaminare più da vicino la struttura algebrica sottostante. Dalla relazione $a^r \equiv 1 \pmod{N}$ segue immediatamente che:

$$a^r - 1 \equiv 0 \pmod{N}$$

Poiché r è pari per ipotesi, possiamo fattorizzare il termine $a^r - 1$ utilizzando la differenza di quadrati[NC10, p. 233]:

$$a^r - 1 = (a^{r/2})^2 - 1 = (a^{r/2} - 1)(a^{r/2} + 1)$$

Ne consegue che:

$$(a^{r/2} - 1)(a^{r/2} + 1) \equiv 0 \pmod{N}$$

Questa congruenza ci dice che N divide il prodotto $(a^{r/2}-1)(a^{r/2}+1)$. Tuttavia, per ipotesi, $a^{r/2} \not\equiv -1 \pmod{N}$, il che significa che $a^{r/2} + 1$ non è divisibile per N. Analogamente, poiché r è il periodo minimo, anche $a^{r/2} - 1$ non può essere divisibile per N (altrimenti $r/2$ sarebbe il periodo). Di conseguenza, N divide il prodotto di due fattori senza dividere ciascun fattore singolarmente. Questo implica necessariamente che ciascuno dei due fattori condivide almeno un divisore primo con N, ma non tutti[NC10, Appendice 4.3, Teorema 5.2]. Il calcolo del massimo comun divisore permette quindi di estrarre questi divisori non banali.

3.2.3 Esempio numerico completo

Per rendere concreti i concetti esposti, consideriamo un esempio numerico completo. Supponiamo di voler fattorizzare $N = 15$. Scegliamo $a = 2$ e verifichiamo che $\gcd(2, 15) = 1$, come richiesto[IBM24]. Procediamo quindi al calcolo della sequenza di potenze di a modulo N:

x	$2^x \bmod 15$
1	2
2	4
3	8
4	1

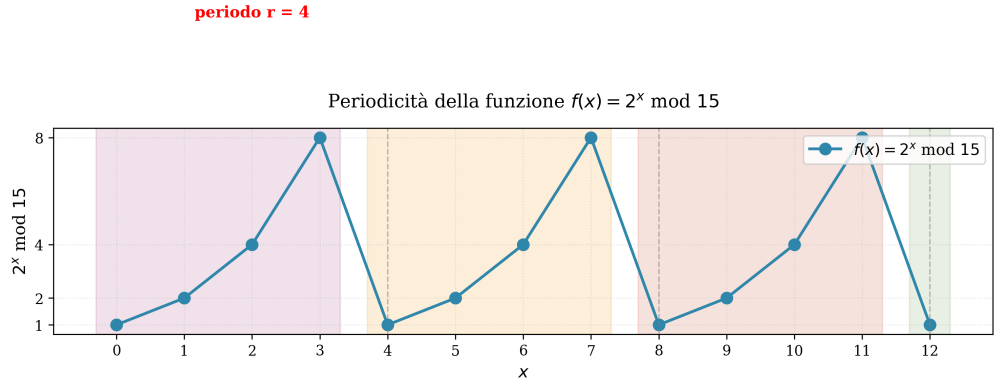

Figure 3.2: Visualizzazione della periodicità della funzione $f(x) = 2^x \bmod 15$. I colori evidenziano i cicli periodici di lunghezza $r = 4$. Elaborazione propria basata sull'esempio presentato da IBM Quantum Documentation.

Osserviamo che il valore si ripete dopo quattro iterazioni, pertanto il periodo è $r = 4$. Verifichiamo ora le condizioni necessarie: il periodo è effettivamente pari, e calcoliamo $a^{r/2} = 2^2 = 4$. Poiché $4 \not\equiv -1 \pmod{15}$ (infatti $4 \equiv 4 \pmod{15}$), entrambe le condizioni sono soddisfatte[NC10, Box 5.4]. Possiamo quindi applicare la formula per estrarre i fattori:

$$\gcd(4 - 1, 15) = \gcd(3, 15) = 3$$
$$\gcd(4 + 1, 15) = \gcd(5, 15) = 5$$

Abbiamo così ottenuto la fattorizzazione $15 = 3 \cdot 5$.

La complessità computazionale del metodo

A questo punto è importante sottolineare quali passaggi del metodo siano effettivamente computazionalmente trattabili e quale rappresenti invece il collo di bottiglia. I passaggi che non presentano difficoltà computazionali significative includono il calcolo del massimo comun divisore, che può essere eseguito efficientemente tramite l'algoritmo di Euclide[NC10, Sezione A4.2], e l'esecuzione di esponenziazioni modulari, che può essere realizzata in tempo polinomiale mediante l'algoritmo di esponenziazione veloce[NC10, Box 5.2].

Il passaggio critico dell'intero procedimento è la determinazione del periodo r. Nel contesto della computazione classica, il calcolo del periodo della funzione

$a^x \bmod N$ richiede essenzialmente di valutare la funzione ripetutamente fino a quando non si osserva la ripetizione del valore iniziale. Il numero di valutazioni necessarie nel caso peggiore può essere dell'ordine di N, e anche nel caso medio il numero di operazioni cresce in modo tale da rendere il problema intrattabile per valori di N sufficientemente grandi. È precisamente in questo passaggio che risiede il collo di bottiglia computazionale che impedisce alla fattorizzazione classica di essere efficiente.

Casi di fallimento e strategie di gestione

È importante osservare che il metodo descritto non ha sempre successo. Possono verificarsi due situazioni in cui la procedura non produce una fattorizzazione: il periodo r potrebbe risultare dispari, oppure potrebbe accadere che $a^{r/2} \equiv -1 \pmod{N}$. In entrambi questi casi, non è possibile applicare la fattorizzazione attraverso il massimo comun divisore nel modo descritto. Tuttavia, è possibile dimostrare che la probabilità di incorrere in uno di questi casi sfavorevoli è relativamente bassa[NC10, Teorema A4.13]. Inoltre, qualora si verifichi uno di questi casi, è sufficiente scegliere un valore diverso di a e ripetere l'intero procedimento[IBM24]. Con alta probabilità, dopo pochi tentativi si otterrà un periodo che soddisfa le condizioni necessarie.

Sintesi del contributo matematico

Ricapitolando, dal punto di vista strettamente matematico, l'algoritmo di Shor opera attraverso una sequenza logica di trasformazioni: in primo luogo, trasforma il problema della fattorizzazione in un problema di determinazione della periodicità di una funzione; in secondo luogo, sfrutta proprietà fondamentali dell'aritmetica modulare per collegare il periodo alla struttura dei fattori di N; in terzo luogo, riduce il problema al calcolo dell'ordine di un elemento modulo N; infine, estrae i fattori cercati attraverso il calcolo del massimo comun divisore[NC10, Sezioni 5.3.1, 5.3.2, Teorema A4.11].

È fondamentale sottolineare che fino a questo punto non è stato introdotto alcun concetto di natura quantistica. Tutto quanto discusso appartiene al dominio della matematica classica e della teoria dei numeri. Il contributo essenziale della computazione quantistica risiede nella capacità di rendere efficiente il passaggio

centrale, ovvero la determinazione del periodo[NC10, Sezione 5.3.1]. Mentre su un computer classico questo passaggio è computazionalmente intrattabile per numeri di grandi dimensioni, vedremo che un computer quantistico è in grado di eseguirlo in tempo polinomiale[NC10, p. 216], rendendo così l'intero algoritmo efficiente e ponendo una seria minaccia alla sicurezza dei sistemi crittografici basati sulla difficoltà della fattorizzazione.

3.2.4 La trasformata di Fourier quantistica

Per comprendere come un computer quantistico risolva efficientemente il problema della determinazione del periodo, è necessario introdurre uno strumento matematico fondamentale: la trasformata di Fourier[NC10, Sezione 5.1]. L'obiettivo di questa parte dell'analisi è chiarire perché la trasformata di Fourier rappresenti lo strumento naturale per affrontare problemi di periodicità e come la sua versione quantistica costituisca l'elemento centrale dell'algoritmo di Shor. Procederemo costruendo il ragionamento a partire da concetti elementari, senza assumere familiarità con gli aspetti tecnici della trasformata.

Il ruolo della trasformata di Fourier Siamo quindi arrivati al problema centrale: determinare il periodo della funzione $f(x) = a^x \bmod N$. Ricordiamo che il calcolo dei singoli valori di $f(x)$ non presenta difficoltà computazionali: dato un valore specifico di x, è possibile calcolare $f(x)$ in modo efficiente. La difficoltà risiede nella determinazione della frequenza con cui i valori della funzione si ripetono.

La trasformata di Fourier è uno strumento matematico che permette di operare un cambio di prospettiva fondamentale. Concettualmente, essa consente di passare da una descrizione di un segnale o di una funzione nel cosiddetto "dominio del tempo" a una descrizione nel "dominio delle frequenze". Il periodo di una funzione è intrinsecamente una proprietà legata alle frequenze: una funzione periodica con periodo r può essere interpretata come un segnale che oscilla con frequenza fondamentale $1/r$. Questa osservazione costituisce la chiave per comprendere perché la trasformata di Fourier sia lo strumento appropriato per estrarre informazioni sulla periodicità.

Intuizione alla base della trasformata di Fourier Visto che l'ottica di questa ricerca è capire l'idea di fondo che si cela dietro shor, prima di vedere la formulazione matematica precisa, è utile sviluppare un'intuizione del funzionamento della trasformata di Fourier attraverso un esempio più semplice. Consideriamo un segnale periodico, che potrebbe rappresentare un'onda sonora, una vibrazione meccanica o qualsiasi altra grandezza fisica che varia nel tempo secondo un pattern ripetitivo. Nel dominio temporale, osserviamo come il segnale evolve istante per istante, registrando il valore della grandezza in funzione del tempo.

La trasformata di Fourier effettua un'operazione concettualmente diversa: prende il segnale nel dominio temporale e lo scompone in una somma di oscillazioni elementari, ciascuna caratterizzata da una frequenza ben definita. In altre parole, invece di descrivere quando il segnale assume determinati valori, la trasformata descrive quali frequenze compongono il segnale e con quale intensità ciascuna frequenza contribuisce. Il periodo del segnale originale emerge naturalmente da questa rappresentazione in termini di frequenze.

Per rendere questa idea più concreta, si consideri un segnale definito come la somma di due sinusoidi:

$$s(t) = \sin(2\pi t) + \sin(4\pi t)$$

Nel dominio temporale, questo segnale appare come un'onda dalla forma piuttosto irregolare, risultante dalla sovrapposizione delle due componenti. Tuttavia, nel dominio delle frequenze, la trasformata di Fourier rivela immediatamente la struttura sottostante: il segnale è composto da esattamente due componenti, una con frequenza 1 e una con frequenza 2. La trasformata di Fourier serve precisamente a estrarre questa decomposizione in componenti frequenziali, permettendo di identificare le frequenze fondamentali che caratterizzano il segnale.

La trasformata di Fourier discreta Nel contesto dell'algoritmo di Shor, non abbiamo a che fare con segnali continui nel tempo, ma piuttosto con sequenze discrete di valori. La funzione $f(x) = a^x \bmod N$ produce una sequenza di valori interi $f(0), f(1), f(2), \ldots$ che si ripete con periodo r. Per analizzare questo tipo di segnali discreti è necessario utilizzare la trasformata di Fourier discreta, comunemente

indicata con l'acronimo DFT (Discrete Fourier Transform)[NC10, Sezione 5.1].

La trasformata di Fourier discreta opera su una sequenza finita di valori $x_0, x_1, \ldots, x_{N-1}$ e la trasforma in un'altra sequenza $X_0, X_1, \ldots, X_{N-1}$, dove ciascun coefficiente X_k misura l'intensità con cui la frequenza k è presente nel segnale originale. La definizione formale della trasformata di Fourier discreta è data da:

$$X_k = \sum_{n=0}^{N-1} x_n \, e^{-2\pi i k n / N}$$

Sebbene non sia necessario memorizzare questa formula nei dettagli, è importante comprenderne il significato concettuale. La sommatoria confronta il segnale originale con tutte le possibili oscillazioni discrete di frequenza k, e il valore di X_k misura quanto bene l'oscillazione di frequenza k si "accorda" con il segnale. Le oscillazioni che sono in fase con il segnale contribuiscono costruttivamente, mentre quelle fuori fase tendono a cancellarsi.

Periodicità e concentrazione spettrale Il collegamento diretto tra periodicità e trasformata di Fourier emerge quando si considera il caso di una funzione perfettamente periodica. Se una funzione possiede periodo r, la sua trasformata di Fourier non è distribuita uniformemente su tutte le frequenze, ma risulta concentrata su valori specifici. In particolare, l'energia spettrale è localizzata in corrispondenza di frequenze che sono multipli interi di $1/r$.

Questa proprietà implica che, osservando il risultato della trasformata di Fourier, è possibile ricavare informazioni precise sul periodo r. I picchi nella trasformata identificano le frequenze caratteristiche, e da queste frequenze si può risalire al periodo. Tuttavia, nel contesto della computazione classica, il calcolo della trasformata di Fourier discreta richiede un numero di operazioni che, anche utilizzando algoritmi efficienti come la FFT (Fast Fourier Transform), cresce almeno proporzionalmente al numero di campioni. Per sequenze di lunghezza comparabile a N, questo rappresenta comunque un costo computazionale significativo che non risolve il problema della complessità della fattorizzazione.

3.2.5 Dalla DFT alla trasformata di Fourier quantistica

La trasformata di Fourier quantistica, comunemente indicata con l'acronimo QFT (Quantum Fourier Transform), rappresenta l'adattamento della trasformata di Fourier discreta al contesto della computazione quantistica[NC10, Sezione 5.2]. È fondamentale comprendere che la QFT non introduce nuove idee matematiche: si tratta essenzialmente della stessa trasformazione matematica definita dalla DFT, ma applicata a stati quantistici invece che a sequenze di numeri classici.

In un computer quantistico, una sequenza di valori viene rappresentata attraverso una sovrapposizione quantistica, ovvero uno stato della forma:

$$\sum_x \alpha_x |x\rangle$$

dove i coefficienti complessi α_x rappresentano le ampiezze di probabilità associate a ciascun valore di base $|x\rangle$. La QFT trasforma questo stato in un nuovo stato:

$$\sum_k \beta_k |k\rangle$$

dove i coefficienti β_k sono esattamente la trasformata di Fourier dei coefficienti originali α_x. In altre parole, la QFT realizza fisicamente, a livello di stato quantistico, la stessa operazione matematica che la DFT realizza a livello di array di numeri.

Il vantaggio computazionale della QFT La ragione per cui la QFT rappresenta un avanzamento rivoluzionario rispetto alla DFT classica risiede in una proprietà fondamentale della computazione quantistica: la capacità di operare simultaneamente su una sovrapposizione di stati[NC10, Sezioni 5.2.1, 5.2.2]. Nel computer quantistico, tutti i coefficienti α_x esistono contemporaneamente nella sovrapposizione, e la QFT agisce sull'intera sovrapposizione in un'unica operazione coerente.

Il risultato di questa applicazione coerente della trasformata è che le ampiezze dei diversi stati interferiscono tra loro secondo le leggi della meccanica quantistica. Le frequenze che sono compatibili con la periodicità della funzione originale vengono amplificate attraverso **interferenza costruttiva**, mentre le frequenze incompatibili

vengono soppresse attraverso **interferenza distruttiva**.

Quando si effettua una misura sullo stato risultante, si ottiene con alta probabilità un valore che contiene informazione sul periodo cercato.

Difatti questo è il principale snodo del teorema, il punto in cui interviene positivamente l'utilizzo dela quantistica, da qui in poi sarà necessario fare altre operazioni per estrarre correttamemnte l'informazione, infatti è importante sottolineare che la misura non fornisce direttamente il periodo r, ma piuttosto un valore dal quale è possibile estrarre r, ma il tutto semplicemente utilizzando tecniche matematiche classiche. In particolare, si utilizza l'algoritmo delle frazioni continue per approssimare il rapporto misurato con una frazione che ha il periodo come denominatore[NC10, Sezione 5.3.3]. Questa fase di post-elaborazione classica ci serve per completare l'algoritmo.

Esempio concettuale del meccanismo Per rendere più concreto il meccanismo di estrazione del periodo, consideriamo un esempio semplificato. Supponiamo che la funzione analizzata abbia periodo $r = 4$ e che il registro quantistico abbia dimensione Q. Dopo l'applicazione della QFT, lo stato quantistico non è distribuito uniformemente, ma presenta concentrazioni di ampiezza in corrispondenza di valori approssimabili come:

$$k \approx \frac{m}{4} \cdot Q$$

dove m è un intero. Quando si effettua la misura, si ottiene con alta probabilità uno di questi valori di k. Dal rapporto:

$$\frac{k}{Q} \approx \frac{m}{r}$$

è possibile ricostruire il periodo r attraverso l'approssimazione con frazioni continue[IBM24]. Questo esempio illustra il collegamento diretto tra l'applicazione della QFT, l'estrazione di informazione sulla periodicità attraverso la misura quantistica, e il recupero finale del periodo che permette la fattorizzazione.

La natura del vantaggio quantistico Avendo capito il principio di funzionamento di shor possiamo chiarire meglio un aspetto concettuale che spesso genera confusione. Il computer quantistico non ottiene il suo vantaggio semplicemente

"provando tutte le frequenze in parallelo" né "calcolando la trasformata esplicitamente come farebbe una CPU classica". Il meccanismo è qualitativamente diverso: il computer quantistico prepara uno stato quantistico con una struttura specifica, sfrutta le proprietà di interferenza e sovrapposizione per far emergere le frequenze rilevanti, e produce il periodo come risultato di un processo probabilistico governato dalle leggi della meccanica quantistica.

La QFT è il meccanismo matematico che rende possibile questa interferenza controllata. Essa permette di organizzare le ampiezze quantistiche in modo tale che, al momento della misura, i valori che contengono informazione sul periodo abbiano probabilità di osservazione significativamente maggiore rispetto agli altri. Questa capacità di manipolare coerentemente le ampiezze di probabilità attraverso interferenza quantistica è ciò che consente al computer quantistico di ottenere un vantaggio esponenziale rispetto ai metodi classici.

3.3 Considerazioni finali

Riassumendo brevemente,

- abbiamo introdotto i concetti di meccanica quantistica necessari a comprendere il funzionamento di un computer quantistico;
- abbiamo convertito il problema della fattorizzazione in un problema di determinazione del periodo di una funzione esponenziale modulare;
- abbiamo poi trovato un algoritmo per calcolarne il perido, e l'abbiamo ottimizzato sostituendo la DFT (che costituiva il contributo con complessità computazionalemnte non polinomiale) con la QFT per ottenere un vantaggio esponenziale rispetto ai metodi classici, permettendo così di risolvere il problema della fattorizzazione in tempo polinomiale su un computer quantistico.

Ora che conosciamo il funzionamento matematico/teorico dell'algoritmo di Shor, possiamo passare alla prossima fare della trattazione: analizzare la risposta della comunità crittografica a questa minaccia: la crittografia post-quantistica.

Bibliography

[AAC+22] Gorjan Alagic, Daniel Apon, David Cooper, Quynh Dang, Thinh Dang, John Kelsey, Jacob Lichtinger, Yi-Kai Liu, Carl Miller, Dustin Moody, et al. Status report on the third round of the nist post-quantum cryptography standardization process. NIST Interagency Report NISTIR 8413, National Institute of Standards and Technology, 2022.

[BBF+19] Nina Bindel, Jacqueline Brendel, Marc Fischlin, Brian Goncalves, and Douglas Stebila. Transitioning to a quantum-resistant public key infrastructure. In *International Conference on Post-Quantum Cryptography*, pages 384–405. Springer, 2019.

[Ber09] Daniel J Bernstein. Introduction to post-quantum cryptography. In *Post-Quantum Cryptography*, pages 1–14. Springer, 2009.

[BL17] Daniel J Bernstein and Tanja Lange. Post-quantum cryptography. *Nature*, 549(7671):188–194, 2017.

[DH76] Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[IBM24] IBM Quantum. Shor's algorithm. IBM Quantum Learning, 2024. Accesso: 2025.

[KKY+19] Philip Krantz, Morten Kjaergaard, Fei Yan, Terry P Orlando, Simon Gustavsson, and William D Oliver. A quantum engineer's guide to superconducting qubits. *Applied Physics Reviews*, 6(2):021318, 2019.

[Kob87] Neal Koblitz. Elliptic curve cryptosystems. *Mathematics of Computation*, 48(177):203–209, 1987.

[LL93] Arjen K Lenstra and Hendrik W Lenstra. *The Development of the Number Field Sieve*, volume 1554 of *Lecture Notes in Mathematics*. Springer, 1993.

[Mil85] Victor S Miller. Use of elliptic curves in cryptography. In *Conference on the Theory and Application of Cryptographic Techniques*, pages 417–426. Springer, 1985.

[Mos18] Michele Mosca. Cybersecurity in an era with quantum computers: Will we be ready? *IEEE Security & Privacy*, 16(5):38–41, 2018.

[Nat20] National Institute of Standards and Technology. Recommendation for key management: Part 1 – general. Special Publication 800-57 Part 1 Rev. 5, NIST, 2020.

[NC10] Michael A Nielsen and Isaac L Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 10th anniversary edition, 2010.

[Pol78] John M Pollard. Monte carlo methods for index computation (mod p). *Mathematics of Computation*, 32(143):918–924, 1978.

[Pre18] John Preskill. Quantum computing in the nisq era and beyond. *Quantum*, 2:79, 2018.

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM*, 56(6):1–40, 2009.

[RSA78] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[Sho94] Peter W Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134. IEEE, 1994.

Acknowledgements

Optional. Max 1 page.