Informatica quantistica: introduzione con esempi in linguaggio C

Francesco e Valentina Sisini

23 giugno 2021

Un prodotto di: Scuola Sisini

https://www.facebook.com/scuolasisini

Scuola Sisini è parte dei SisiniPazzi il cui indirizzo è isisinipazzi.it Prima pubblicazione in Italia 2020

Edizioni: i Sisini Pazzi, 2020

Questo testo è dedicato a Carlo Ferrario (1949-2015), relatore della mia tesi di Laurea.
Per la sua lucidità di pensiero e chiarezza espositiva è stato un punto di riferimento per diverse generazioni di studenti e studentesse che si sono formati presso l'università degli studi di Ferrara.
Per negligenza, non l'ho ringraziato allora. Lo faccio adesso.

Francesco

Scuola Sisini ringrazia calorosamente Lauro Galtarossa per la rilettura, non specializzata ma condotta in forma amichevole, del testo.

Esiste una forbice tra richiesta ed offerta di conoscenza: da un lato articoli scientifici destinati solo ad un pubblico iper specialistico, dall'altro la divulgazione nozionistica di contenuti affascinanti, descritti però qualitativamente.

Tra questi due estremi si inserisce il metodo di divulgazione di Scuola Sisini che, attraverso un percorso ragionato, porta ad uscire dalla propria zona di comfort per far propri gli strumenti e le basi che permettono poi di approfondire autonomamente gli argomenti di interesse.

Indice

Introduzione

L'informatica quantistica si è impegnata in una promessa importante: usare i processi della fisica quantistica per codificare, elaborare e trasmettere informazioni.

Rispetto alle tecnologie informatiche *classiche*, la scienza quantistica propone tre importanti novità non previste nella *fisica/informatica classiche* perché basate sul concetto quantistico di sovrapposizione degli stati ed in particolare su quello di entanglement:

- principio di non clonazione
- teletrasporto quantistico (dell'informazione)
- dense coding (codificazione a maggior densità)

La sfida è duplice: da un lato creare un sistema di trasmissione delle informazioni sicuro e a prova di qualsiasi intrusione, dall'altro scrivere algoritmi che risolvano problemi che attualmente richiedono anni di computazioni anche ai super computer di ultima generazione.
Da sola, la promessa di questi risultati è già sufficiente a far

entusiasmare le grosse compagnie informatiche del momento e a spingerle ad investire ingenti capitali. Ma l'aspetto realmente affascinante della computazione quantistica, che per tanti darà finalmente un senso allo studio della termodinamica compiuto alle scuole superiori o all'università, è che essa è reversibile e, in linea di principio, è eco sostenibile: potrebbe non consumare energia.

Come è organizzato questo libro

Questo testo propone un percorso figurato attraverso idee, concetti, nozioni e dimostrazioni. Raggiungere la fine del percorso è senz'altro importante, ma non bisogna trascurare il valore del percorso in sé che ha l'obiettivo di creare collegamenti e connessioni tra argomenti diversi che qui si fondono in un filo conduttore che porta infine alla capacità di comprendere, e poi di padroneggiare, la nuova scienza informatica che da più di quarant'anni bussa alle porte e che per ora è stata accessibile solo a pochi.
Per stimolare l'interesse, la curiosità e facilitare la comprensione, anche intuitiva, di questo argomento ancora nuovo, il percorso proposto non sarà sempre lineare, ma arricchito di esempi e anticipazioni che hanno lo scopo di introdurre gradualmente gli argomenti lasciando il tempo necessario affinché essi siano assimilati prima di venire trattati con gli strumenti formali necessari. Si potranno quindi incontrare concetti e

termini che vengono inizialmente solo accennati per essere poi ripresi e sviluppati nel seguito del testo.

Questo testo è nato per essere letto in modo completo ed, essendo un'introduzione, predilige la scorrevolezza della lettura allo sviluppo esaustivo degli argomenti trattati.

- Nel capitolo 1 vengono introdotte le prime idee della meccanica quantistica e si percorre rapidamente lo sviluppo del pensiero che ha portato all'ipotesi di poter costruire un computer quantistico.

- Nel capitolo 2 si introducono in modo *informale* i concetti di qubits e di sovrapposizione degli stati. Scopo del capitolo è portare l'attenzione sui veri obiettivi del libro creando l'interesse per proseguirne la lettura e superare il capitolo 3.

- Il capitolo 3 è il più complesso dell'intero testo, ma, dopo averlo compreso, si avranno tutti gli strumenti per capire l'informatica quantistica ad ogni livello. Qui vengono introdotti i principi base della meccanica quantistica necessari per comprendere a pieno il senso del resto della lettura. I principi vengono trattati anche matematicamente. Si raccomanda la lettura dell'intero capitolo.

- La relazione tra reversibilità della computazione e la meccanica quantistica è trattata nel capitolo 4. Questo capitolo rappresenta il cuore dell'argomento e, senza eccedere nei formalismi matematici, presenta un aspetto affascinante delle tecnologie quantistiche emergenti.

- Nel capitolo 5 vengono presentati i gates quantistici. Si concretizzano i concetti visti nel capitolo 3 in componenti circuitali che sono capaci di trasformare lo stato dei qubits da 1 a 0 e da 0 ad 1, analogamente a quanto fanno le porte logiche (logics gates) con i bits *classici*.

- Il principio di non clonazione, il teletrasporto e il dense coding sono presentati e spiegati in dettaglio nel capitolo 6. Fondamenti su cui si basa la potenzialità dell'informatica quantistica, capiti i quali, saranno acquisiti gli strumenti intellettuali per entrare consapevolmente nello studio e nella pratica della computazione quantistica.

- Nel capitolo 7 vengono discussi i principi alla base della realizzazione del computer quantistico. Gli elementi presentati sono aggiornati allo stato dell'arte del 2021.

- Nel capitolo 8 viene presentato un esempio concreto di algoritmo che sfrutta il prinicio di parallelismo intrinseco della computazione quantistica. Il codice presentato può essere provato su un simulatore o sui server quantistici messi a disposizione in cloud come, ad esempio, quello di IBM.

- Nell'appemdice, vengono proposti una serie di esercizi da eseguire per fissare le idee sui concetti appresi. Questi sono esercizi di programmazione in linguaggio C che per essere svolti richiedono di aver compreso le nozioni presentate nei capitoli precedenti.

Al termine del capitolo sono presentate le soluzioni dei problemi proposti.

A chi è rivolto questo testo?

Questo libro può essere letto in chiavi diverse a seconda della propria preparazione di base.

Completamente digiuno di fisica e di informatica Lo yoga ci insegna che il primo movimento della respirazione deve essere l'espirazione, perché nei polmoni serve spazio prima di inspirare nuovo ossigeno. Quindi, se vi trovate nella situazione di non sapere nulla, non disperate, perché allora c'è posto per il nuovo.

Sazio di informatica ma digiuno di meccanica quantistica Se la teoria di Shannon e il concetto di entropia dell'informazione sono già note, la loro applicazione nell'ambito quantistico sarà frutto di sorpresa e soddisfazione. In questo caso potrete apprezzare come concetti già noti possono essere rivisitati in un'ottica completamente nuova.

Sazio di fisica ma digiuno di informatica È difficile completare il corso di studi in fisica senza aver scritto qualche riga di codice, ma si sa che la programmazione è solo una piccola fetta dell'informatica e quindi non disperate perché troverete nel testo le informazioni essenziali per applicare le vostre cono-

scenze di meccanica quantistica alla teoria informatica e, dove il testo deve glissare, troverete i giusti riferimenti per completare lo studio in modo indipendente.

Esperto di fisica e di informatica La fisica e l'informatica si sono trovate già a stretto contatto nei primi anni '40. Ora, di nuovo, è la fisica che ha la responsabilità di guidare lo sviluppo tecnologico fino a rendere l'informatica quantistica una realtà ingegnerizzata. Questo testo ha l'ambizione di fornire una prima guida introduttiva per mettere in relazione la meccanica quantistica e la teoria dell'informazione.

Convenzioni tipografiche

Nel testo sono inseriti dei **Box** che contengono degli approfondimenti sull'argomento trattato nel paragrafo. I box sono segnalati dalla scritta **Box**, incorniciati e con lo sfondo giallo.

Nel testo si è fatto uso del **grassetto** per evidenziare i termini chiave in una frase e del *corsivo* per le parole che hanno un significato specifico nel contesto del discorso.
Per esempio:

La meccanica quantistica inserisce all'interno della teoria anche il concetto di **misura**. Si deve notare che anche la teoria classica prevedeva le misure sperimentali...

Per programmare un computer *classico* è necessario prendere confidenza con la tecnologia: tastiera, monitor, editor ecc., e apprendere il concetto di *linguaggio formale* con il quale è possibile descrivere un algoritmo.

Pagina web del libro

Sul sito https://pumar.it è presente la pagina web del libro (https://pumar.it/libroQuantistica.php)
Sulla pagina si possono trovare diversi link di interesse e il link con gli **errata corrige** che emergeranno nel corso del tempo.
Sono state programmate una serie di revisioni del testo con scadenza mensile, gli eventuali errori rilevati saranno riportati e corretti. I codici C presentati nel capitolo 7 sono disponibili all'indirizzo:
https://github.com/francescosisini/ nel repository:
LIBRO-Informatica-Quantistica
Gli argomenti sono inoltre trattati sul canale di ScuolaSisini:
https://www.youtube.com/channel/UCDwLlFqa0xZ71PEOdHA0aSQ

Premessa

Questo testo è una prima introduzione ai concetti e ai principi dell'informatica quantistica. Qui sono presentati e spiegati tutti gli strumenti matematici, fisici e informatici necessari per una comprensione completa e consapevole. Il testo è auto consistente e, per una prima lettura, non richiede l'ausilio di altri testi specifici. Per mettere in pratica i concetti presentati alla fine del libro sono inseriti in un capitolo apposito degli esercizi di programmazione in linguaggio C. Il capitolo può essere saltato senza pregiudicare la comprensione del resto.

CAPITOLO 1

Punto di partenza

La fisica è una scienza impegnativa perché non si accontenta di risposte verosimili, ma pretende una continua verifica sperimentale di ogni affermazione che viene fatta in suo nome. Nonostante questo obiettivo è ironico sapere che ogni teoria è probabilmente falsa, e quelle che crediamo vere, sono in attesa di essere dimostrate false o appunto *falsificate*.
Con questo, si deve rinunciare al proposito di studiarle? No certo, perché le teorie della fisica, entro certi limiti, funzionano.
Nel XX secolo è stato dimostrato che la teoria della gravitazione universale è falsa, ciò nonostante, essa è ancora molto utile per numerosi calcoli di balistica spaziale e per i conti più *grossolani* di astronomia del sistema solare.
La teoria di Newton sulla gravitazione è stata sostituita da quella di Einstein della relatività generale. Negli ultimi decenni pe-

rò anche quest'ultima ha dato segni di cedimento e in futuro potrebbe essere messa in discussione.

Le leggi che la fisica identifica come leggi fondamentali costituiscono la *meccanica* del nostro universo.
Il primo incontro *scolastico* che si ha con la meccanica è la *cinematica*. Questa descrive le relazioni tra posizione e velocità di un *punto materiale*. Il concetto di punto materiale è un concetto *classico* che parte dall'idea che la materia possa essere pensata in modo analogo alla geometria euclidea, dove i solidi hanno superficie e volume definiti, ma sono costituiti da insiemi di punti ognuno dei quali ha dimensione nulla.
Allo stesso modo il punto materiale ha dimensione nulla, ma ad esso è associata una massa.
Lo stato del punto materiale è completamente descritto quando sono definite le sue tre coordinate spaziali x, y e z e le tre componenti cartesiane della sua velocità: v_x, v_y e v_z.
La posizione e la velocità del punto materiale possono variare con il tempo e per questo si specifica che sia le coordinate che la velocità sono funzioni di quest'ultimo: $x(t)$, $y(t)$ e $z(t)$ e $v_x(t)$, $v_y(t)$ e $v_z(t)$.

La meccanica quantistica trova il suo primo **disaccordo** con la meccanica classica già qui: secondo la meccanica quantistica non è possibile che ad un dato istante di tempo, per una particella, si conoscano **esattamente** la sua posizione e la sua velocità. Questo è noto come *principio di indeterminazione di Heisenberg*.

Come è stato chiarito all'inizio del paragrafo, la meccanica

quantistica si discosta notevolmente dalla meccanica classica, ma non è definibile senza di essa.
Nella meccanica classica un sistema fisico si ritiene completamente definito quando siano note tutte le posizioni dei punti che lo compongono, tutte le velocità e le eventuali forze che stanno agendo. Ovviamente non può essere lo stesso per un sistema quanto-meccanico, visto che queste grandezze non possono essere note simultaneamente con precisione arbitraria.
La meccanica quantistica inserisce all'interno della teoria anche il concetto di **misura**. Si deve notare che anche la teoria classica prevedeva le misure sperimentali, anzi queste fanno parte del metodo sperimentale con cui la teoria è stata costruita, ma nella meccanica classica tali misure sono intese come esterne alla teoria.

La differenza tra i due approcci è che nel *classico* il processo di misura è visto come un'azione necessaria all'osservatore dell'esperimento per conoscere empiricamente il valore di una grandezza che sta osservando, ma che detta grandezza abbia un ben dato valore indipendente dal processo di misura. Nel *quantistico* invece, secondo l'interpretazione della scuola di Copenaghen, le grandezze fisiche di un dato sistema non hanno sempre valori definiti, ma li assumono nel momento stesso in cui accade un evento *speciale*, come appunto una misura.

A questo proposito è d'obbligo chiarire subito per evitare malintesi. Il concetto di **misura** in meccanica quantistica non prevede la presenza di un essere intelligente: con misura si intende (per esempio) l'interazione di un sistema molto piccolo, cioè di

dimensioni atomiche, sub-atomiche o particellare con un sistema classico, cioè di grande massa. Non si pensi che grande massa significhi la massa di un edificio, basta prendere venti grammi di acqua per avere (circa) un numero di Avogadro (6×1023) di molecole, quindi il concetto di grande massa va rapportato alla scala atomica (Landau, 1947; Bohr 1928).

Questo aspetto della teoria è difficile da capire perché si scontra contro l'esperienza quotidiana che per l'appunto è un'esperienza *classica* della fisica.

Domanda n. 1 Il principio di indeterminazione di Heisenberg riguarda:

1. L'impossibilità di misurare simultaneamente e con precisione arbitraria la posizione e la quantità di moto di una particella

2. L'impossibilità di predire correttamente l'evoluzione dello stato di un sistema fisico

1.1 Mondo classico e mondo quantistico: la doppia fenditura

Se lasciamo cadere un oggetto dalla mano, durante la sua caduta, siamo convinti che esso abbia una posizione ed una velocità ben definite.
Non importa se lo stiamo osservando o meno: secondo il senso

comune l'oggetto si trova sempre in uno stato fisico ben definito.
Le cose stanno diversamente secondo la meccanica quantistica (Tonomura, 1989). Per fare un esempio famigliare, consideriamo un vecchio televisore a tubi catodici e proviamo a seguire il percorso di un elettrone che fuoriesce dal catodo. Supponiamo di porre tra il catodo e lo schermo televisivo una targhetta in metallo con due fenditure vicine, in modo che se l'elettrone non azzecca una delle due venga fermato dalla targhetta.
Dopo aver acceso la TV, vederemo comparire dei puntini luminosi sullo schermo secondo una data disposizione, e penseremo:

ogni puntino sullo schermo corrisponde ad un elettrone che è riuscito ad attraversare una fenditura

Nel nostro pensiero, che segue l'intuizione classica, un elettrone che è riuscito a passare deve aver attraversato l'una o l'altra fenditura della targhetta.
Secondo la meccanica quantistica, invece, le cose non sono andate esattamente così.
Se durante il moto dell'elettrone non è stato condotto alcun esperimento per misurare la sua posizione, allora non si può dire se l'elettrone abbia seguito una specifica traiettoria e neanche se sia passato dall'una o dall'altra fenditura, ma si può dire solo che:

se è arrivato sullo schermo, allora è passato per l'una e/o per l'altra fenditura.

A prima vista questa considerazione può sembrare priva di

interesse pratico, quasi un sofisma, ma dal 1980 in avanti ha assunto un ruolo chiave nella tecnologia futura, cioè da quando il fisico Richard Feynman ha pensato che la meccanica quantistica poteva essere la base per la costruzione di un nuovo modello di calcolo, differente da quello sviluppato da Alan Turing (Feynman, 1982).
Questo aspetto della natura è chiamato principio di sovrapposizione degli stati.
Da esso discendono un teorema e due importanti applicazioni dell'informatica quantistica: il teorema del **no cloning principle** e le due applicazioni **dense coding** e **quantum teleportation**, che saranno discusse dettagliatamente più avanti nel testo.

1.2 Meccanica e computazione quantistica

Per comprendere la meccanica quantistica servono almeno tre anni di studio: uno dedicato alle basi matematiche, partendo dal calcolo differenziale in una sola dimensione e sul campo dei numeri reali, per arrivare al calcolo sul campo dei numeri complessi; uno dedicato allo studio della meccanica hamiltoniana ed uno alla meccanica quantistica, teoria non relativistica e teoria relativistica.
Questo studio **non può** essere condotto dopo cena come passatempo, ma deve impegnare il giorno intero.

Per fortuna si possono comprendere i principi dell'informatica quantistica e la base del funzionamento di un computer quantistico anche senza conoscere tutta la meccanica quantistica.
La situazione presente non è diversa dallo scenario in cui furono presentati i primi computer nel secolo scorso.
Le prime macchine erano la sintesi perfetta dello stato dell'arte della logica matematica e dell'elettronica, che allora era una scienza su cui avevano competenza solo i fisici perché i corsi di ingegneria elettronica ancora non esistevano.
Negli anni '50 del XX secolo per spiegare il funzionamento di un computer era necessario un fisico esperto e per scrivere un programma serviva comunque un matematico o un ingegnere.
Il concetto di memoria volatile era assolutamente nuovo e si basava su tecnologie al limite del realizzabile, per cui era difficile separare l'idea astratta di **automa** dalla sua concreta realizzazione.
Come risultato di questo scenario la programmazione di un computer appariva possibile solo ad esperti fisici, matematici ed ingegneri.
Anche senza correre troppo indietro negli anni, basta pensare al film "War Games" di John Badham (1983). Il film racconta la storia di un giovane hacker che si trova ad affrontare un'intelligenza artificiale che scambia la guerra mondiale per un gioco di strategia.
È interessante notare che il programma di intelligenza artificiale venga presentato come il frutto del lavoro di un unico scien-

ziato: Stephen Falken.
L'idea dello *scienziato* capace da solo di realizzare *un mostro* come quello del dottor Frankenstein, era un retaggio della prima informatica degli anni '50.

Dalla visione di un'informatica lontana ed *aliena* si è passati in pochi decenni a linguaggi di programmazione come *scratch* pensati per introdurre la programmazione già dalle scuole elementari.
Oggi è possibile formare un programmatore in pochi mesi senza che egli conosca nulla di elettronica.
Per programmare un computer classico è necessario prendere confidenza con la tecnologia: tastiera, monitor, editor ecc., e apprendere il concetto di *linguaggio formale* con il quale è possibile descrivere un algoritmo, anche senza sapere come funziona effettivamente un computer.

Quanto appena detto per la programmazione di computer classici, vale anche per i computer quantistici. Se si conosce la meccanica quantistica si può capire a fondo come funzioni la loro tecnologia, ma la comprensione approfondita di questa non è necessaria per formulare un algoritmo quantistico.
Il messaggio quindi è il seguente: con un certo sforzo, chiunque si applichi, può comprendere e applicare la computazione quantistica e, se studierà in modo completo la meccanica quantistica, comprenderà anche come funziona un computer quantistico.
Non mi sembra nulla di diverso dallo scenario attuale.

In questo testo svilupperemo diversi esempi di programmi

che simuleranno alcuni aspetti di un computer quantistico. Va da sé che la simulazione non potrà riprodurre le prestazioni in termini di velocità.

Domanda n. 2 Quali sono i tre aspetti peculiari della informatica quantistica di cui si è accennato?:

1. Dense coding, quantum teleportation, indeterminazione degli stati
2. Dense coding, quantum teleportation, no cloning principle
3. Dense coding, no cloning principle, sovrapposizione degli stati

1.3 Simulare la fisica sul computer: si può fare Mr. Feynman?

Il titolo di questo paragrafo nasce dalla parafrasi del titolo del libro "Sta scherzando Mr. Feynman?" di Ralph Leighton, amico di Feynman, che raccolse diversi aneddoti sullo scienziato. Richard Feynman era un fisico con una ricca carriera accademica che aveva lavorato con J. von Neumann a metà degli anni '40 e vide nascere i primi computer *classici*.

L'idea di un modello di computazione basato sulla meccanica quantistica risale almeno al 1980, proprio per opera di Feynman, che presentò la sua idea nel 1981 con un articolo

intitolato Simulating physics with computers, pubblicato sulla rivista International Journal of Theoretical Physics, nel quale poneva una domanda precisa:

È possibile simulare la fisica quantistica usando un computer?

Nell'articolo, Feynman spiegava le ragioni per cui i computer classici erano adatti a simulare la fisica classica, ma non la fisica quantistica.

È interessante notare l'influenza che von Neumann deve avere avuto sull'allora giovane Feynman, nell'articolo, infatti, Feynman si riferisce ai computer sempre in termini di automi cellulari, idea nata appunto dalla mente di von Neumann mentre cercava di progettare un modello di automa che potesse *auto riprodursi*.

L'argomentazione usata da Feynman, di seguito esposta, è importante perché ci introduce direttamente ai principi di meccanica quantistica che ci sono necessari per sviluppare il resto del testo, in modo particolare al principio di *località* della fisica classica.

1.3.1 La fisica classica è locale

La fisica classica è una *teoria locale*, cioè: l'interazione tra due *entità fisiche* avviene solo quando sono a diretto contatto.
Questa affermazione potrebbe sembrare in contrasto con i fenomeni elettromagnetici ed elettrostatici: pensiamo per esempio a due punti materiali che esercitano l'uno sull'altro una forza coulombiana.

In realtà anche l'elettromagnetismo è una teoria locale e infatti l'azione a distanza tra due cariche è dovuta al *campo di forze* indotto da ogni carica che agisce sull'altra carica, quindi l'azione su ogni carica è comunque sempre **locale** e quindi dipende solo dal valore del campo nel punto in cui si trova la carica.

Per chiarire bene il proprio punto di vista Feynman fece riferimento al modello computazionale dell'**automa cellulare**.
A differenza della più nota *architettura di von Neumann*, dovuta naturalmente a von Neumann, l'automa cellulare ha la caratteristica che ogni bit cambia il proprio valore in base al valore che hanno i bit adiacenti, quindi l'*interazione è locale*. In realtà anche i computer odierni che si basano sull'architettura di von Neumann sono *locali*, ma nell'automa cellulare questa *località* è resa più evidente, e per questo Feynman preferì riferirsi ad essa.

Box: Modello di calcolo dell'automa cellulare
Questo modello è stato introdotto da von Neumann che, studiando i primi modelli di intelligenza artificiale nel campo informatico, pensò all'automa cellulare per realizzare una forma di automa che potesse auto riprodursi.
La particolarità dell'automa cellulare è che la computazione avviene in modo *parallelo* e puntuale o, appunto, **locale**. L'automa infatti consiste in:

- Una stringa formata di bit che possono essere nello stato 0 oppure 1. Ogni bit è detto cella.

- Una regola che stabilisce come ogni cella cambi di stato in base al valore delle celle vicine.

Il sistema di calcolo è molto semplice. All'istante $t = 0$ l'automa si trova in una certa configurazione iniziale. All'istante $t = 1$, per ogni cella, si valutano due bit: quello a destra e quello a sinistra della cella e, in base alla regola stabilita, si assegna il nuovo valore del bit per la configurazione dell'automa a $t = 1$. Si ripete la procedura per i t successivi.

Per fare un esempio di si consideri la seguente configurazione:

- Configurazione iniziale: 11011011
- Regola:
  - $1\mathbf{1}1 \rightarrow \mathbf{0}$
  - $1\mathbf{1}0 \rightarrow \mathbf{1}$
  - $1\mathbf{0}1 \rightarrow \mathbf{1}$
  - $1\mathbf{0}0 \rightarrow \mathbf{0}$
  - $0\mathbf{1}1 \rightarrow \mathbf{1}$
  - $0\mathbf{1}0 \rightarrow \mathbf{1}$
  - $0\mathbf{0}1 \rightarrow \mathbf{1}$
  - $0\mathbf{0}0 \rightarrow \mathbf{0}$

La computazione dei primi 4 passi è rappresentata nella tabella seguente.

$t = 0$	(1)	1	1	0	1	1	0	1	1	(1)

$t=1$	(0)	0	1	1	1	1	1	1	0	(0)
$t=2$	(0)	1	1	0	0	0	0	1	0	(1)
$t=3$	(1)	1	1	0	0	0	1	1	1	(1)

I bit fra parentesi sono usati per il *padding*, cioè per poter racchiudere tra due celle anche le celle dei bordi. In questo esempio il padding è ottenuto per *circolarità*: in pratica è come se la stringa di bit si chiudesse su sé stessa. Cambiando le regole è possibile definire diversi tipi di automi che possono anche essere definiti su una griglia a due dimensioni. Quello qui mostrato è noto con il nome di **Rule 110**: un automa Turing completo che in linea di principio può essere usato per eseguire un qualsiasi programma che può essere eseguito su un calcolatore ordinario.

Tornando al punto, Feynman illustra come sfruttando la *località* del computer sia possibile **simulare** la fisica classica ma, al contrario, **non** sia possibile simulare la fisica regolata dalla meccanica quantistica.

Sia chiaro che in un computer classico può essere eseguita l'evoluzione di un sistema quantistico: se si conosce la dinamica di un sistema quantistico, cioè le equazioni che descrivono la sua evoluzione nel tempo (l'analogo delle leggi orarie delle coordinate nella fisica classica), è assolutamente possibile **emularlo**, ma non simularlo, perché la meccanica quantistica non è locale. Questo argomento è molto complesso e può dar luogo a fraintendimenti, infatti la non-località della meccanica quantistica non significa che per essa sia ammessa un'azione

a distanza, ma che sia possibile riscontrare una correlazione tra misure sperimentali che non è spiegabile da nessuna teoria locale. La non località della meccanica quantistica è legata al concetto di entanglement che viene discusso nei prossimi capitoli.

È importante avere presente questo argomento perché è stato uno delle motivazioni principali che ha spinto Feynman e altri insieme e dopo di lui ad indagare la possibilità di realizzare una macchina quantistica che permettesse di simulare la fisica quantistica come i computer ordinari permettono di simulare la fisica classica.

Domanda n. 3 In base al principio di *località* della fisica classica, come si spiegano i fenomeni elettrostatici?:

1. Si spiegano con la presenza di fotoni virtuali, quindi con la meccanica quantistica
2. Inserendo il campo elettrico come una entità fisica
3. Sono una eccezione alla teoria

CAPITOLO 2

Dai bits classici ai qubits quantistici

In questo capitolo vengono esposti i principi della meccanica quantistica che devono essere appresi e *digeriti* per continuare con successo la lettura del testo.
Invece di formulare i principi e i postulati con l'ambizione di descrivere l'intero mondo fisico, verranno formulati con l'obiettivo di descrivere ciò che serve a comprendere la computazione quantistica.
In questo capitolo introdurremo questi concetti in modo informale, cercando di abituarci alle nuove idee che devono essere apprese. Nel capitolo successivo daremo una visione più formale e compatta degli stessi concetti.

2.1 I bits nell'informatica classica

Un bit è l'unità di informazione usata in informatica classica. Un bit può assumere solo due valori: l'1 e lo 0. Usando più bit si possono rappresentare delle informazioni complesse, per esempio il numero decimale 11 può essere rappresentato in forma binaria come la sequenza di bit 1011 che sviluppata sulle potenze del due risulta esattamente il numero decimale 11 (i.e. $1 \times 2^0 + 1 \times 2^1 + 0 \times 2^2 + 1 \times 2^3 = 1 + 2 + 8$).

Un bit può essere *realizzato* con diverse tecnologie. In generale si tratta di creare un **sistema fisico** che ammetta due possibili **stati** di equilibrio. Chiameremo questo sistema: **sistema fisico classico** dove l'aggettivo classico si riferisce alla possibilità di descrivere la fisica del sistema usando le leggi della meccanica classica (Newton, per intenderci).

Per esempio si può pensare ad un bit realizzato da una matita in cui il valore 1 è associato allo **stato** in cui la matita è in posizione verticale, e il valore 0 allo **stato** in cui la matita è in orizzontale.
Lo stato *verticale* della matita può essere espresso matematicamente indicando con θ l'angolo tra la matita e il piano (per esempio il piano del tavolo). Se θ è uguale a $\frac{\pi}{2}$ allora il bit è 1, se θ è zero allora il bit è 0.
In pratica possiamo associare il valore *binario* del bit allo stato θ del sistema fisico che, nel caso in questione, è una semplice matita.

Ci sono diversi altri modi per realizzare un bit, per esempio

usando un interruttore collegato ad una luce led, e associando alla luce accesa il valore 1 e alla luce spenta lo 0.
Ovviamente per condensare molti giga bits in uno spazio limitato non si possono usare matite o led, ma si usano altre tecnologie come quelle presenti nei moderni calcolatori.

2.1.1 Proprietà dei bits

In questo paragrafo ci soffermiamo su alcune osservazioni che possono sembrare ovvie, ma che acquisiranno interesse nel momento in cui le confronteremo con osservazioni analoghe compiute sui **qubit** cioè i bit quantistici.
Per semplicità continuiamo a considerare i bit identificati dallo stato θ della matita, ma le considerazione che trarremo valgono in generale per qualsiasi tecnologia *classica* che si possa usare per realizzarli.

2.1.2 Relazione tra lo stato di un bit e il suo valore

Lo stato di un *bit-matita* è sempre perfettamente determinato.
La matita può stare solo nello stato $\theta = \frac{\pi}{2}$ o nello stato $\theta = 0$ e, a causa della forza di gravità, non è possibile che si trovi in equilibrio uno stato intermedio.
Per conoscere il valore del bit è sufficiente conoscere il suo angolo rispetto al piano.

Se un osservatore, sulla Terra, si trovasse ruotato rispetto al piano di un angolo α vedrebbe il *bit-matita* formare un angolo

diverso dai due angoli possibili appena definiti: dal punto di vista dell'osservatore, infatti, la matita si troverebbe ad uno dei due angoli $\frac{\pi}{2} + \alpha$ o $0 + \alpha$.

Questa rotazione non impedirebbe di ricondurre la posizione del *bit-matita* al corretto valore del bit, infatti, l'osservatore saprebbe di essere ruotato perché potrebbe verificarlo con un esperimento legato alla forza gravitazionale: per esempio osservando la direzione in cui gli oggetti cadono a terra. In tal modo potrebbe sempre ricondurre la propria misurazione dell'angolo formato dalla matita nel suo sistema di riferimento, con l'angolo θ usato per definire lo stato della matita.

Quindi, per i bit *classici*, possiamo affermare che:

Lo stato di un bit è associato sempre ad uno ed un solo valore: 0 oppure 1.

2.1.3 Modifica dello stato di un bit

Lo stato di un *bit-matita* può essere modificato solo in seguito ad una azione volontaria che lo porti dallo stato $\theta = 0$ a $\theta = \frac{\pi}{2}$ oppure da $\theta\frac{\pi}{2}$ a $\theta = 0$.

Domanda n. 4 In linea di principio, per un computer *classico*, il risultato della lettura di un bit dalla memoria è prevedibile?:

1. Si, fatta eccezione per possibili problemi hardware del dispositivo
2. No, è un fenomeno probabilistico
3. Si, ma solo se si è eseguita una copia di back-up

2.1.4 Copia dello stato di un bit

Lo stato di un bit è sempre noto ed è possibile eseguirne una copia senza modificare l'originale.
Questa caratteristica permette molte operazioni utili come la copia di un documento da condividere o da inviare per email, e altrettante delicate o rischiose dal punto di vista della sicurezza informatica come per esempio un intruso che può leggere un messaggio scambiato tra due soggetti comunicanti senza che loro se ne accorgano intercettando la comunicazione e copiandone il contenuto bit a bit, senza compromettere il messaggio trasmesso.

2.1.5 Stato di più bits

Il bit è l'unità minima di informazione dell'informatica (discreta). Per rappresentare delle grandezze complesse, come ad esempio per codificare un alfabeto o le cifre decimali, un singolo bit non è sufficiente e quindi se ne devono usare un certo insieme in cui ogni bit a seconda della sua posizione abbia un significato preciso, come visto nell'esempio della trasformazione della sequenza binaria 1101 nel numero decimale 11.

Consideriamo per esempio un insieme di otto *bit-matita* (byte) disposti alcuni nello stato $\theta = 0$ e altri nello stato $\theta = \frac{\pi}{2}$.
Per riferirci in modo chiaro allo stato di ognuna delle matite useremo l'indice i. Ad esempio scrivendo θ_0 indicheremo lo stato della prima matita, mentre scrivendo θ_7 quello dell'ottava.
Usando queste otto matite creiamo un nuovo sistema fisico che

ammette 2^8 configurazioni diverse e che corrispondono ad una precisa sequenza dei valori θ_i.
Per esempio lo stato $\begin{pmatrix} \pi/2 & 0 & 0 & 0 & 0 & 0 & 0 & \pi/2 \end{pmatrix}$ corrisponde ai bits $\begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}$.
La corrispondenza individuata è di natura biunivoca e quindi alle 256 diverse configurazioni corrispondono esattamente 256 diverse informazioni codificate dagli otto bits, non di più e non di meno.

Abbiamo portato volutamente l'attenzione sulla relazione tra l'unità di informazione (1 e 0) e il concetto di sistema fisico classico, perché nel prossimo paragrafo riprendiamo questa relazione, applicandola però al concetto di sistema fisico **quantistico**.

2.2 I qubits

Ci chiediamo ora se esiste una differenza tra sistemi fisici classici e sistemi quantistici.
A rigore di logica dovremmo dire che la teoria quantistica è considerata la meccanica che regola ogni sistema fisico esistente, quindi esistono solo sistemi quantistici.
In realtà è prassi indicare come quantistici solo quei sistemi che non possono essere spiegati con sufficiente accuratezza dalla meccanica classica.
Per esempio, per descrivere il classico moto di una palla di cannone, lanciata ad un angolo 0 di inclinazione, non abbiamo bisogno della meccanica quantistica, e chiamiamo un sistema

siffatto *classico*.
Chiamiamo quindi sistemi quantistici quei sistemi fisici che richiedono la meccanica quantistica per essere descritti correttamente.
Sistemi come gli atomi e le particelle elementari hanno sempre natura quantistica, ma anche certi sistemi *macroscopici*, normalmente non presenti in natura sulla Terra, ma ottenuti per via artificiale, possono essere sistemi quantistici se il loro comportamento non è descrivibile dalla fisica classica e richiede l'uso della meccanica quantistica.

Realizzare un esperimento per misurare un sistema quantistico non è semplice come realizzarlo per un sistema classico.

Gli esperimenti per acquisire delle misure classiche, come ad esempio la posizione di un oggetto o il tempo di percorrenza di un carrello lungo un binario, sono facilmente realizzabili, e per questo fanno parte del bagaglio di studio ed esperienza che si acquisisce già frequentando la scuola.
Per questo il concetto di **stato** introdotto con l'esempio della matita è diretto ed intuitivo, in quanto rientra nelle esperienza di tutti.

Il bit è una associazione tra uno stato fisico classico e uno dei due valori numerici 0 e 1.
Analogamente, il qubit è un'associazione tra uno stato fisico quantistico e uno dei due valori numerici 0 e 1. La differenza esistente tra stato fisico classico e quello quantistico è ciò che determina la differenza tra l'informatica classica e quella

quantistica.

Il concetto di **stato** che definiremo per i sistemi fisici quantistici non è invece altrettanto intuitivo di quello classico perché, nella vita comune, mancano le occasioni per familiarizzare con esperimenti di misura quantistica.

Per definire i qubits, nel seguito di questo capitolo, introdurremo le particelle chiamate **fotoni** perché useremo la misura del loro stato fisico associandogli il valore binario 0 o 1.

Domanda n. 5 Cosa si intende con sistema quantistico?:

1. Un sistema di dimensioni atomiche, cioè paragonabili a quelle degli atomi

2. Qualsiasi sistema, anche macroscopico, che *richieda* di essere descritto con le leggi della meccanica quantistica, come ad esempio certi sistemi macroscopici a temperature prossime agli 0 gradi Kelvin

3. Un sistema che non può essere misurato con strumenti classici

2.3 I fotoni: le particelle del campo elettromagnetico

I fotoni vengono introdotti in fisica teorica in seguito ad un procedimento matematico noto come *seconda quantizzazione*.
La prima quantizzazione è un procedimento con cui si ottiene la descrizione quantistica a partire dalla descrizione classica di

un sistema fisico. La prima quantizzazione è applicata per descrivere le particelle come ad esempio l'elettrone, il positrone, ecc.

La seconda quantizzazione si applica invece alle entità fisiche note come *campi*: ad esempio il campo elettromagnetico. Dal procedimento matematico di quantizzazione, descritto per esempio da Dirac (Dirac, 1957), si individuano delle soluzioni matematiche che hanno una interpretazione fisica molto interessante.

Ogni *onda piana* del campo elettromagnetico, può essere vista come un insieme di particelle prive di massa ma con energia, quantità di moto e polarizzazione definite. Queste particelle sono dette **fotoni**. La fisica sperimentale aveva evidenziato già dall'inizio del XX secolo la natura *quantizzata* dello scambio di energia tra atomi e campo elettromagnetico: Einstein ricevette il premio Nobel proprio per aver descritto in modo efficace il meccanismo con cui il campo elettromagnetico estrae gli elettroni dagli atomi cedendo energia in quantità discreta e non continua come invece ci si sarebbe aspettato dalla natura continua del campo elettromagnetico. Questa quantità discreta di energia è associabile all'energia rilasciata da un fotone assorbito dall'atomo: il noto *effetto fotoelettrico*.

Per questo motivo i fotoni, che sono stati ottenuti come procedura matematica dal campo elettromagnetico, hanno guadagnato rapidamente *credibilità* come particelle, e oggi si tende a considerarli *reali* tanto quanto gli elettroni, i protoni, ecc.

2.3.1 Come è fatto un fotone se non ha massa?

Non c'è una risposta scientifica a questa domanda perché la meccanica quantistica rinuncia alla descrizione *pittorica* di particelle e campi e si limita a descrivere ciò che succede nel corso di un esperimento senza aggiungere elementi non misurabili dalla teoria, e quindi, in meccanica quantistica, un fotone *è in sé la descrizione* di come il campo elettromagnetico si propaga ed interagisce con gli altri elementi fisici come elettroni, positroni, ecc., a prescindere da cosa esso possa essere veramente. Viene la tentazione di immaginare i fotoni come piccole particelle che trasportano l'energia elettromagnetica viaggiando alla velocità della luce.
È importante precisare che ai fini della comprensione della informatica quantistica, figurarsi il fotone non è necessario, anzi potrebbe risultare controproducente.
Purtroppo la meccanica quantistica non è una teoria intuitiva e i tentativi di renderla tale possono creare confusione.

2.3.2 Stato di un fotone

Il qubit è associato allo stato di un sistema quantistico. Ci sono diversi possibili sistemi fisici che possono essere usati per realizzare i qubits ma, il fotone è sicuramente il più semplice da capire e, probabilmente, quello candidato ad essere più usato per le realizzazioni tecnologiche integrate in sistemi di dimensioni ridotte. La caratteristica del fotone è che può essere usato sia come qubit per la computazione, sia come qubit

per la trasmissione di dati. Cosa più complessa quando i qubit sono realizzati come stati di sistemi aventi massa che non possono essere trasmessi da un punto all'altro senza movimento di materia.

Un fotone trasporta energia, non ha massa e *viaggia* alla velocità della luce nel vuoto. Oltre a queste proprietà ne ha un'altra, ereditata dal campo elettromagnetico, molto importante perché è quella che useremo per stabilire il suo **stato**, la direzione di oscillazione del campo elettrico, cioè la **polarizzazione**.

Per capire cosa sia la polarizzazione del fotone è istruttivo partire dalle onde elettromagnetiche descritte nella teoria classica. Cosa sono le onde elettromagnetiche?
Un esempio di fenomeno fisico descritto da onde elettromagnetiche è la luce visibile. L'occhio animale vede grazie all'interazione di una certa *gamma* di onde elettromagnetiche con i neuroni presenti nella retina dell'occhio. La luce è quindi descritta da onde elettromagnetiche e può essere polarizzata.
Vediamo ora due esempi in cui si presentano degli effetti dovuti alla polarizzazione della luce.
È esperienza comune che osservando il cielo si vedano le stelle brillare, mentre la Luna e i pianeti visibili, appaiano più opachi e non brillanti. Questo fenomeno è dovuto alla riflessione della luce, che riflettendo sulla superficie di questi astri rimane polarizzata.
Un altro fenomeno simile, dovuto alla polarizzazione, si osserva indossando un paio di occhiali con filtro polaroid. Questi

occhiali sono in grado di eliminare i riflessi perché permettono solo alla luce polarizzata lungo una certa direzione di attraversali.

La polarizzazione della luce è la direzione che ha il campo elettrico dell'onda elettromagnetica.

Per un onda elettromagnetica piana, tale direzione si modifica seguendo una legge precisa del tempo, per esempio ruotando con velocità costante. Nel caso più semplice si mantiene costante, e in questo caso si parla di **polarizzazione lineare**.
La polarizzazione della particella fotone discende direttamente dalla polarizzazione delle onde elettromagnetiche.
La particella fotone eredita, dalle onde elettromagnetiche della fisica classica, la direzione di propagazione e la polarizzazione, mentre la sua energia è legata alla frequenza ω di oscillazione del campo elettrico (uguale a quella del magnetico) dalla relazione $\mathbf{E} = \mathbf{h}\omega$, dove h è una costante detta di Planck.

Negli esempi da qui in avanti, i fotoni saranno le nostre *matite quantistiche*, che come i *bit-matita* saranno usati per definire il valore dei **qubits**

La polarizzazione di un fotone è descritta usando due soli stati, esattamente come è per la matita. Nel caso classico della matita abbiamo visto che i due stati sono definiti come $\theta = \frac{\pi}{2}$ e $\theta = 0$, per i fotoni useremo una notazione diversa, indicheremo i due stati come $|0\rangle$ e $|1\rangle$

Domanda n. 6 Cosa lega le onde elettromagnetiche piane dell'elettrodinamica classica ai *fotoni* della teoria quantistica?:

1. I fotoni hanno la stessa polarizzazione e direzione di propagazione delle corrispettive onde elettromagnetiche

2. Ogni fotone ha la stessa intensità del campo elettrico e magnetico dell'onda elettromagnetica

3. Il fotone è un fenomeno ottico, mentre le onde elettromagnetiche hanno natura elettrica e magnetica

2.4 Fotoni e qubits

In questo paragrafo vediamo di concretizzare maggiormente il concetto di qubit e di capire come realizzarlo usando un sistema fisico quantistico.

Abbiamo visto che, in un sistema classico, i due possibili stati della matita sono un esempio di bit, similmente:

lo spazio di tutti i possibili stati in cui può trovarsi la polarizzazione di un fotone, è un esempio di qubit.

La relazione tra i due stati, verticale e orizzontale, di una matita e i due valori 1 e 0 del bit è chiara. Presa una matita, il valore del bit associato ad essa è ottenuto misurando l'angolo che la matita forma rispetto ad un certo piano. Una volta stabilita la convenzione che la matita verticale significa 1 e orizzontale significa 0, non è possibile alcun fraintendimento tra diversi possibili soggetti che effettuano misure sulle matite, perché la direzione verticale è definita usando una direzione preferenziale, cioè quella in cui agisce la forza di gravità.
Per i fotoni invece non abbiamo un fenomeno importante come

la gravità per definire una direzione preferenziale perché la loro polarizzazione non ne è influenzata.

Tra qubits e bits c'è una importante analogia: entrambi sono associati a sistemi fisici caratterizzati da due soli stati, nel seguito vedremo però che ci sono anche delle differenze molto importanti dovute al diverso significato attribuito alla parola *stato* nella fisica classica e in quella quantistica.

Nella storiella che segue vediamo come utilizzare lo stato quantico della polarizzazione di un fotone per associarvi un qubit cioè l'analogo quantistico del bit che possiamo chiamare scherzosamente *bit-fotone*. Questa serve a dare una visione d'insieme dei concetti presentati fin'ora ed è necessaria per avvicinarsi con gradualità allo studio della meccanica quantistica che viene approfondito nel capitolo 3.

2.5 Il signor Rossi e il signor Ferrari: una classica storia di bits

Il signor Rossi è una persona a modo abituata a non distinguersi troppo dagli altri. Il mattino quando arriva al lavoro, dispone le sue matite per formare un codice binario che verrà letto pochi minuti dopo dal signor Ferrari, suo collega.
Al contrario di Rossi, Ferrari è un tipo eccentrico e ama farsi notare, per questo invece di camminare come fanno tutti gli altri, gira attaccato a due pertiche mantenendo una posizione obliqua rispetto al suolo. Ama vedere il mondo da un altro punto di vista.

Quando il mattino si reca presso la scrivania del signor Rossi per leggere i bit delle sue matite, non si lascia confondere dal proprio punto di vista perché sa che deve giudicare lo stato dei bit a partire dalla direzione in cui penzola la sua cravatta!

2.6 Il signor Magri e il signor Fabbri: una storia di qubits

Il signor Magri produce molti dati tutti i giorni, e non avendo modo di registrarli tutti in un archivio, deve inviarli a chi ne ha bisogno: il signor Fabbri, poi eliminarli. Il signor Magri abita lontano dal signor Fabbri. Ogni mattina si assicura di preparare un dispositivo polarizzatore per trasmettere dei fotoni polarizzati al signor Fabbri.
Il signor Magri è una persona scrupolosa e prepara i fotoni in modo che essi vengano prodotti con polarizzazione verticale $|1\rangle$ oppure orizzontale $|0\rangle$.
Con polarizzazione verticale ($|1\rangle$), egli intende un fotone che attraversi (senza essere fermato) un filtro *polaroid* con asse di polarizzazione disposto verticalmente, cioè nella direzione in cui agisce la gravità, mentre con polarizzazione orizzontale ($|0\rangle$) un fotone che attraversi un polaroid con asse orizzontale. Egli chiama l'insieme formato da $|1\rangle$ e $|0\rangle$ **base standard** per la polarizzazione dei fotoni.

2.6.1 Il primo incidente: dati recuperabili

Il signor Fabbri ogni mattina attende i fotoni del signor Magri, con i quali egli deve comunicargli informazioni importanti. Egli sa che i fotoni possono avere solo due possibili polarizzazioni però ha frainteso l'accordo con il signor Magri, e ha sistemato gli assi di polarizzazione dei suoi polaroid in modo errato, infatti li ha invertiti tra di loro, scambiando in questo modo i fotoni con polarizzazione $|1\rangle$ con quelli a polarizzazione $|0\rangle$ e di conseguenza i qubit a 1 con i qubit a 0.

Un giorno per puro caso, il signor Fabbri e i signor Magri si sentono al telefono e ne approfittano per discutere dei dati che il signor Magri ha appena inviato al signor Fabbri e che non ha ancora cancellato. I due si rendono conto che qualcosa non va, e presto si accorgono che tutti i valori dei qubit sono diversi.
Il signor Fabbri ha l'archivio con tutti i dati trasmessi dall'inizio dal sig Magri e questi lo tranquillizza dicendogli che sarà sufficiente trasformare tutti i qubit che sono a 0 al valore 1 e quelli che sono ad 1 al valore 0.

2.6.2 Il secondo incidente: dati non recuperabili

Dopo altre due settimane di trasmissione dei dati secondo la stessa procedura, il signor Fabbri si accorge che inavvertitamente ha ruotato di 45 gradi entrambi gli assi dei polarizzatori. Nonostante questo disallineamento il suo sistema ha sempre funzionato, continuando a registrare i due possibili valori 0 o

1.
Per sua fortuna, il signor Fabbri, riesce a risalire al momento dell'*incidente* che capisce essere avvenuto durante le pulizie del suo laboratorio.
Forte della esperienza già avuta dal primo incidente, egli ripone fiducia nell'esistenza di una legge fisica o informatica che permetta di risalire ai valori corretti dei qubit partendo dai valori registrati con i polarizzatori ruotati di 45 gradi in senso orario. Egli però non conosce questa legge e si rivolge al signor Magri scoprendo suo malgrado che neanche lui la conosce, ma di nuovo ha tenuto copia dei dati trasmessi.
Da un confronto degli ultimi dati trasmessi con quelli ricevuti, si accorge che non c'è correlazione, infatti i fotoni trasmessi come $|1\rangle$ sono stati ricevuti a volte come $|1\rangle$ e altre come $|0\rangle$ in modo apparentemente casuale e con la stessa frequenza di casi. Il signor Magri e il signor Fabbri capiscono che non c'è modo di risalire ai dati corretti partendo dai dati registrati.
È finita bene anche questa volta grazie alla scrupolosità del signor Magri che aveva conservato per qualche giorno in più i dati. Però i due si sono accorti che il loro sistema di trasmissione dei dati nasconde qualcosa di particolare.

2.6.3 Un'osservazione interessante

Il signor Fabbri è molto amareggiato e si auto accusa di negligenza. Il signor Magri invece è rimasto colpito dell'accaduto, e decide di sperimentarlo ulteriormente. Chiede allora al signor Fabbri di ruotare volontariamente i due polaroid, ma di un an-

golo di soli 30 gradi. Ciò fatto, egli trasmette un serie di fotoni di cui archivia il valore per confrontarli con quelli ricevuti dal signor Fabbri.
Dal confronto segue un risultato inaspettato: questa volta esiste una relazione tra i qubit ricevuti e quelli inviati, però è solo una relazione statistica: i qubit ricevuti sono per il 75% dei casi corretti, e per il restante 25% errati.

2.7 I signori Magri, Fabbri, Rossi e Ferrari si incontrano

I quattro signori Magri, Fabbri, Rossi e Ferrari si incontrano e si raccontano le relative esperienze. Essi comprendono che l'utilizzo della posizione delle matite per rappresentare i bit è assai diverso dall'utilizzo della polarizzazione dei fotoni.

Il signor Magri vuole provare a capire cosa sta succedendo con i fotoni e prova ad impostare una serie di ragionamenti partendo da quanto ha osservato fin'ora:

- un fotone verticale ($|1\rangle$) passa per un polaroid polarizzato in modo verticale
- un fotone orizzontale ($|0\rangle$) passa per un polaroid polarizzato in modo orizzontale
- un fotone verticale è fermato da un polaroid orizzontale
- un fotone orizzontale è fermato da un polaroid verticale

Desidera esprimere il fatto che un fotone polarizzato verticalmente ha il 50% di probabilità di passare attraverso un polaroid inclinato di 45 gradi, come ha capito dal secondo incidente occorso.

Sapendo che nell'universo non esiste una direzione preferenziale, pensa che, l'effetto di un fotone, a polarizzazione verticale su un polaroid obliquo (ruotato di 45 gradi in senso antiorario), debba essere lo stesso di un fotone a polarizzazione obliqua (45 gradi in senso orario) su un polaroid verticale. In pratica, decide di ruotare la polarizzazione del fotone rispetto l'asse del polaroid anziché fare il contrario: perché la cosa importante è l'angolo fra i due e non l'angolo rispetto la verticale, visto che la polarizzazione non è influenzata dalla gravità (come visto nel paragrafo precedente).

Per scrivere i due stati del fotone rispetto al polaroid ruotato egli si figura di dover ruotare nel piano cartesiano un segmento di lunghezza unitaria con centro nell'origine e associa all'asse x lo stato $|0\rangle$ e all'asse y lo stato $|1\rangle$ e scrive la rotazione di 45 gradi in senso orario dello stato $|0\rangle$ di polarizzazione del fotone come:

$$\begin{pmatrix} \frac{1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} |0\rangle - |1\rangle \end{pmatrix}$$

e quella della rotazione di 45 gradi in senso orario dello stato $|1\rangle$ del fotone come:

$$\begin{pmatrix} \frac{1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} |0\rangle + |1\rangle \end{pmatrix}$$

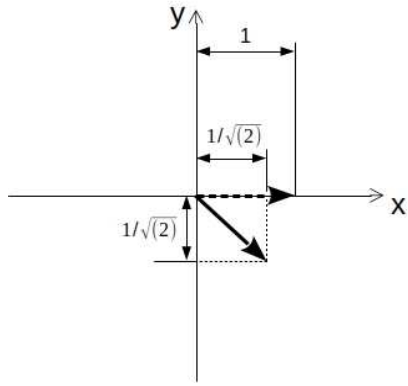


Figura 2.1: Il vettore ha lunghezza unitaria. Dopo la rotazione di 45 gradi, le sue componenti cartesiane misurano entrambe $\frac{1}{\sqrt{2}}$.

Il signor Magri postula che la probabilità p che un fotone polarizzato a 45 gradi lungo la diagonale secondaria (come in figura) passi attraverso un polaroid ad asse verticale è data da:

$$\mathbf{p} = \left(\frac{1}{\sqrt{2}} \right)^2 = \frac{1}{2}$$

Il signor Magri non sa spiegarsi il perché, però capisce che c'è una relazione tra la radice di 2 trovata nella rotazione della polarizzazione del fotone e la probabilità del 50% (appunto $\frac{1}{2}$) misurata sperimentalmente, ma essendo un informatico non è interessato alla polarizzazione dei fotoni in sé, ciò che sta catturando la sua attenzione è, invece, la relazione che hanno i qubit con lo stato di polarizzazione dei fotoni ed è questo che lui vuole studiare.
Prima di continuare i suoi studi capisce che le espressioni tro-

vate sono importanti e decide di assegnare dei simboli agli stati da esse individuati. Sceglie i due simboli $|0'\rangle$ e $|1'\rangle$ e li definisce come segue:

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

e

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

Si accorge quindi che può riscrivere gli stati $|0\rangle e\,|\,1\rangle$ usando gli stati $|0'\rangle$ e $|1'\rangle$. Infatti:

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

e

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

Il signor Magri capisce che i qubit possono essere rappresentati anche attraverso questi due stati e che quindi questi due stati sono una base per rappresentare i qubit. Il signor Magri ha quindi intuito un concetto importante:

lo stato di polarizzazione di un fotone è esprimibile rispetto ad una base formata da due stati separati e indipendenti

ed esso può anche non coincidere con nessuno dei due stati di base, come gli è successo quando i fotoni con polarizzazione obliqua incidevano sui polaroid ad assi verticale ed orizzontale. Il signor Magri ha capito un'altra cosa molto importate:

se la polarizzazione del fotone è parallela a uno degli assi dei polaroid, allora il risultato della misura è certo, altrimenti, se la polarizzazione del fotone giace su una retta non parallela a nessuno dei due assi, allora il risultato è solo probabilistico.

2.8 Differenze tra bits e qubits

La storia del signor Magri e del signor Fabbri, ci ha aperto gli occhi sulle differenze tra bits e qubits.
Entrambi sfruttano lo stato di un sistema fisico per identificare un coppia di valori (1 e 0) e questa è la loro **analogia**. La **differenza** è che per i primi c'è una corrispondenza *biunivoca* tra lo stato e il valore del bit, mentre per i secondi c'è una corrispondenza biunivoca tra la *misura* dello stato e il valore del qubit.

Nel caso classico, lo stato della matita identifica univocamente il valore del bit (verticale 1, orizzontale 0), nel caso quantistico invece stati diversi possono portare probabilisticamente allo stesso valore del bit, quindi non si può stabilire una corrispondenza biunivoca tranne nei casi in cui lo stato di polarizzazione coincide con uno degli assi dei polaroid.
La misura dello stato di polarizzazione, invece, porta sempre ad un risultato che può essere solo uno tra i due possibili e quindi esiste una relazione biunivoca tra la misura dello stato e il valore del qubit.

Questa differenza è molto importante ed è ciò che caratterizza l'informatica quantistica rispetto a quella classica. Quando

si sarà finito di leggere il libro, si torni su questo punto fintanto che non risulti completamente chiaro.

2.8.1 Perché questa differenza?

Nella fisica classica, non c'è differenza tra lo stato di un sistema e la sua misura.
Ripensando all'esempio delle matite si può immaginare di adottare la stessa convenzione a bordo di una **stazione spaziale** orbitante. In quel contesto le matite possono assumere angoli arbitrari rispetto al riferimento della stazione quindi, sebbene una coppia di assi rimanga un valido sistema per definirne lo stato, si potrebbe obiettare che analogamente a quanto accade nella meccanica quantistica, se una matita non è allineata lungo uno dei due assi, il suo stato è in una sovrapposizione di stati.
Questo non è falso, ma c'è una differenza importante e sostanziale con il concetto di stato quantistico. Dal punto di vista classico, anche se sulla stazione spaziale la matita può assumere una qualsiasi angolazione rispetto agli assi, rimane una relazione biunivoca tra stato e misura:

la misura dell'angolo che la matita forma con un dato asse può essere eseguita con precisione arbitraria senza modificare lo stato della matita.

Anche su una stazione spaziale si potrebbero quindi usare i *bit-matita* adottando la semplice convenzione che l'angolo da

0 a 45 gradi rappresenti il bit 0 mentre l'angolo da 45 a 90 rappresenti il bit 1.

Per i sistemi quantistici invece: **la relazione tra lo stato del sistema e il risultato della misura ha natura probabilistica.** Come abbiamo visto, quando si misura lo stato di polarizzazione di un fotone si possono ottenere solo due valori distinti. Prima della misura, il fotone può essere in un infinità di stati diversi tra loro. Il processo di misura *obbliga* il fotone ad assumere uno dei due stati che per questo sono detti *stati di base*. Questo aspetto della meccanica quantistica è difficile da digerire ed in effetti esistono anche interpretazioni diverse da questa, detta *interpretazione di Copenaghen*, che offrono una visione alternativa. Quella di Copenaghen è comunque l'interpretazione più diffusa e per questo è detta essere *ortodossa*.

2.9 Stato fisico del fotone e valore del qubit

Alcuni testi potrebbero tendere ad unificare il concetto di qubit con quello di stato fisico quantistico della polarizzazione del fotone. Indipendentemente dal modo in cui lo si vuole trattare formalmente, è sempre importante tenere a mente la distinzione esistente tra lo stato fisico prima e dopo l'esecuzione di una misura.

Ora che abbiamo introdotto il concetto di stato quantistico dobbiamo chiarire bene qual'è la relazione **tra stato fisico** e **valore del qubit**.

Dato un fotone, si è visto che, ad esso è associato un qubit legato al suo stato di polarizzazione che possiamo indicare con la lettera ψ o con il simbolo $|\psi\rangle$ come ha fatto il signor Magri.
Dal concetto di bit, ci aspettiamo che al qubit sia associato un valore binario, o lo zero oppure l'uno.
Il valore binario associato al qubit non è il suo stato $|\psi\rangle$ in sé, quindi ci chiediamo in che forma sia ad esso legato.
Per determinare il valore di un qubit è necessario sempre compiere una **misura**, nel caso in questione una misura della sua polarizzazione.
La misura *obbliga* lo stato (o funzione d'onda) ψ a *collassare* in uno dei due stati che rappresentano la base scelta per eseguire la misura stessa.
Supponendo di aver eseguito la misura rispetto ai due assi dei polaroid, dopo la misura, lo stato $|\psi'\rangle$ del fotone potrà essere:

- $|\psi'\rangle$= $|0\rangle$ oppure
- $|\psi'\rangle$= $|1\rangle$

e questo **determina** il valore del qubit ad essere rispettivamente o 0 oppure 1.
Lo stato di un qubit può essere rappresentato graficamente sulla *sfera di Bloch*, un sistema di mappatura per i sistemi quantistici a due livelli (vedi fig. 2.2). Questo argomento avanzato è trattato in modo approfondito in *Qubits: principi fondamentali* degli stessi autori.

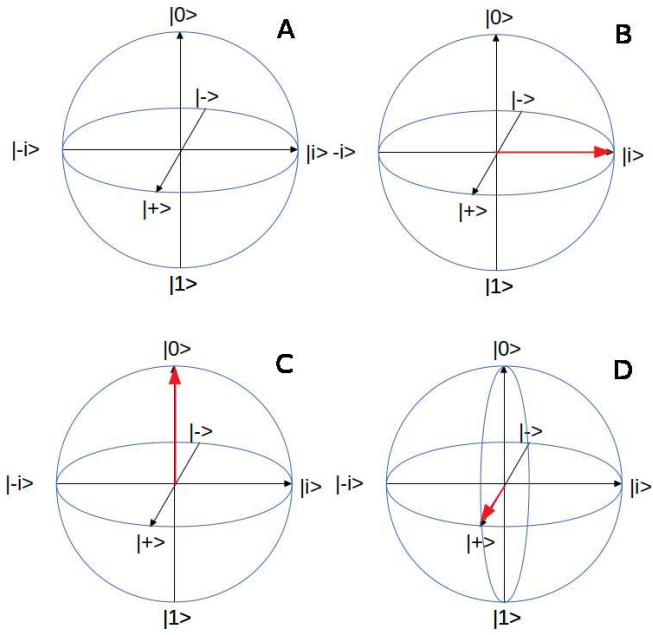


*Figura 2.2: La figura mostra la sfera di Bloch, un sistema grafico per rappresentare lo stato di un qubit (**A**).*
Il vettore che giace lungo il semiasse positivo indica che il qubit è nello stato $|i\rangle = \frac{1}{\sqrt{(2)}}(|0\rangle + i|1\rangle)$ *(**B**). Il qubit è nello stato* $|0\rangle$ *(**C**). Il qubit è nello stato* $|+\rangle = \frac{1}{\sqrt{(2)}}(|0\rangle + |1\rangle)$ *(**D**).*

Domanda n. 7 La principale differenza tra i *bit-matita* e i *bit-fotoni* è che:

1. Per i primi c'è una relazione deterministica tra il loro stato e la misura del loro valore, mentre per i secondi no
2. I *bit-matita*, sebbene non efficienti, sono implementabili realmente, i secondi sono solo un ipotesi teorica
3. Non mostrano differenze e per questo è possibile realizzare concretamente dispositivi che sfruttano la meccanica quantistica per la computazione

CAPITOLO 3

Principi della meccanica quantistica

In questo capitolo si pongono le basi canoniche per comprendere i principi della meccanica quantistica. I concetti presentati in modo informale e discorsivo nel capitolo precedente, sono qui ripresi dall'inizio e sviluppati fin dove necessario agli obiettivi del testo.

Per comprendere ed *accettare* la formulazione della meccanica quantistica, così come si presenta oggi, è importante partire dalla visione di chi l'ha concretamente formulata, cioè P.A.M Dirac (Dirac, 1957).

Agli orecchi di chi non è un fisico di professione, questo potrebbe sembrare strano: *in fondo si tratta di formule che o sono giuste o sono sbagliate!*

Non è così perché la fisica è la risposta alle domande che ci si

pone e la sua forma dipende dalle domande stesse e dal modo in cui queste sono state poste.
Per questo studiare le risposte della fisica, senza conoscere il pensiero di chi l'ha formulata, può essere molto complesso, specialmente se la formulazione non è intuitiva, come appunto il caso della meccanica quantistica.

Dirac scrive che l'obiettivo principale della fisica non è quello di fornire una raffigurazione della natura, ma di fornire le leggi che governano i fenomeni conosciuti e che possono guidare alla scoperta di fenomeni nuovi.
Si noti che Dirac scrive esattamente:

. . . the main object of physical science is not the provision of picture,. . .

evidenziando la **non necessità** di raffigurarsi il mondo quantistico.
Dirac aveva un buon motivo per sostenere questo pensiero visto che si era assunto la responsabilità di stravolgere la visione che la fisica aveva avuto fino ai primi anni '20.

Il XX secolo era cominciato con diverse *novità* interessanti per la fisica e la formulazione delle leggi dell'elettromagnetismo aveva aperto le porte a nuove frontiere dello studio e della ricerca. All'inizio del secolo Rutherford ipotizza il modello planetario per gli atomi, de Broglie ipotizza la dualità onda corpuscolo, Planck ipotizza il quanto di azione che diverrà la base per la meccanica quantistica ed Einstein spiega l'effetto fotoelettrico basandosi sull'idea di fotone.
I fisici e le scoperte sarebbero molti di più, ma ci accontentia-

mo di questi appena citati perché ci servono solo per giungere ad altri due: Schrodinger e Heisenberg che furono tra i primi a comprendere che le scoperte e le teorie che avevano salutato il nuovo secolo non potevano trovare una collocazione nel *vecchio* paradigma classico newtoniano e che un nuovo paradigma doveva essere formulato.
Lo fecero entrambi, ognuno a modo suo. Il primo formulò la **meccanica ondulatoria** e il secondo la **meccanica matriciale**.
La prima era più semplice, perché parlava una lingua più conosciuta ai fisici ed ebbe maggior successo iniziale, ma la seconda si rivelò più semplice nel proseguo perché era complessa per trattare i problemi semplici, ma semplice per i problemi difficili. Fu raccolta da Dirac e divenne la **meccanica quantistica**.

Questa breve introduzione ha avuto lo scopo di preparare il lettore digiuno di meccanica quantistica a quanto seguirà, perché la meccanica quantistica può essere apprezzata più facilmente se si comprende a fondo che nasce dal pensiero di altri esseri umani, non è vera o falsa, è una teoria, un modo di pensare e di sperimentare, che si sta rivelando molto utile, tanto quanto lo è stata la meccanica newtoniana prima di lei.

Domanda n. 8 Chi è considerato il *padre* della meccanica ondulatoria?

1. Dirac
2. Schrodinger
3. Heisenberg

3.1 Formulazione matematica del principio di sovrapposizione

Dobbiamo ora addentrarci nei dettagli della teoria quantistica. Nella trattazione che segue si è cercato di estrapolare solo il necessario della teoria ai fini dell'introduzione all'informatica quantistica, attenendosi però rigorosamente al testo di Dirac da cui nasce la principale linea di interpretazione della meccanica quantistica, quella detta della *Scuola di Copenaghen*.

Il lettore sia rassicurato: non è necessario comprendere tutto e perfettamente, ma piuttosto sia determinato a leggere questi capitoli per intero, in modo da non trovarsi sprovveduto nei capitoli dedicati all'informatica dove i concetti e le idee qui esposte troveranno applicazione.
Non si saltino poi le domande e ci si fermi a ragionarci sopra, consultando nel caso le risposte in fondo al testo.

3.2 Stato fisico

Consideriamo un sistema atomico composto di corpi (puntiformi) ognuno con una specifica massa, carica elettrica, ecc. I corpi interagiscono tra loro attraendosi o respingendosi secondo certe leggi della fisica.
Ci saranno vari possibili moti per ognuno di questi corpi che saranno in accordo con le leggi della fisica. Seguendo Dirac, chiameremo ognuno di questi possibili moti *stato* del sistema.
Nella meccanica classica lo stato di un sistema può essere com-

pletamente definito misurando le tre coordinate x, y, z e le tre componenti v_x, v_y e v_z della velocità di ognuno dei corpi che costituisce il sistema. Questo risulta possibile perché la meccanica classica è stata storicamente applicata a sistemi di *grande massa* che coinvolgono sempre, almeno, decine di migliaia di atomi.
La meccanica quantistica si applica ai sistemi composti anche da pochi o singoli atomi e alle particelle elementari. A tali dimensioni si ha che:

i corpi non posseggono simultaneamente una posizione x ed una quantità di moto p definita con precisione, ma esiste una incertezza minima data da $\Delta x \Delta p_x \geq \hbar$.

Quello appena enunciato è noto come il principio di indeterminazione di Heisenberg.
A causa di questa incertezza, non è possibile definire il concetto di traiettoria per i corpi come atomi ed elettroni e per questo motivo il concetto di stato fisico **non può essere preso in prestito** dalla meccanica classica, ma:

lo stato di un sistema *piccolo* (quantistico) deve essere specificato usando meno variabili rispetto uno classico o usando dei dati che sono più indefiniti. (Dirac)

3.3 Probabilità e misura nella meccanica quantistica

Sia la meccanica ondulatoria che quella matriciale introducono un'ulteriore novità rispetto alla *picture* della natura che era stata indotta dalla meccanica classica.
In meccanica classica conoscere lo stato di un sistema fisico significa poter prevedere il risultato di una misura.
Si consideri ad esempio il sistema fisico costituito dalle sponde di un biliardo e dalle palle libere di rotolare sul tappeto. Se ad un certo istante sono note le coordinate x, y e la velocità v_x, v_y di ogni pallina, **questo implica** che il risultato di un esperimento che misurasse la posizione e la velocità delle palle darebbe come risultato esattamente le posizioni x, y e le velocità v_x, v_y note del sistema.
Nella meccanica classica questo è ovvio in quanto, in linea di principio, non esiste differenza tra lo stato delle variabili di un sistema e le misure delle variabili del sistema, visto che le misure possono essere eseguite con precisione arbitraria.
Nel mondo atomico le cose sono descritte diversamente.
Quando si esegue una misura il risultato dipende dallo stato, ma in modo **probabilistico**.
La cosa può sembrare inconsueta: ha lasciato perplessi molti fisici del passato e crea dubbi anche in molti del presente. Questo approccio alla fisica comunque porta a risultati coerenti con la sperimentazione e inoltre ha permesso di sviluppare molte

nuove tecnologie, come appunto la computazione quantistica e quindi vale la pena provare a capirlo.

Domanda n. 9 Quali grandezze devono essere conosciute per caratterizzare completamente lo stato di un corpo (puntiforme) nella meccanica classica?

1. Velocità e forza

2. Posizione e accelerazione

3. Posizione e velocità

3.4 Sovrapposizione di stati

Attenzione: in questo capitolo non verrà usata subito la notazione $|\psi\rangle$ per indicare uno stato quantistico già introdotta nel paragrafo precedente, perché l'argomento viene qui ripreso e spiegato dall'inizio in modo canonico.

Come si è costruita una fisica in cui i risultati delle misure sono probabili anziché certi?
L'idea sviluppata nella meccanica quantistica è semplice quanto non intuitiva. Si accetta che ogni stato fisico sia la sovrapposizione di altri stati fisici e che quindi un sistema sia sempre in una sovrapposizione di stati.

3.5 Idea classica

Per fare un esempio pensiamo alle matite del signor Ferrari viste nel capitolo precedente. Dal punto di vista di Ferrari esse sono oblique e quindi sono in una sovrapposizione dello stato verticale e di quello orizzontale.
Rispetto al suolo, le matite possono essere rappresentate come:

$$bit\,0 \rightarrow \hat{\mathbf{i}}l$$
$$bit\,1 \rightarrow \hat{\mathbf{j}}l$$

dove l è la lunghezza della matita e $\hat{\mathbf{i}}$ e $\hat{\mathbf{j}}$ rappresentano i vettori direzionali dell'asse x e dell'asse y rispettivamente.
Dal punto di vista del signor Ferrari, che le guarda ruotato di $\frac{\pi}{4}$ (45 gradi in senso orario), la relazione tra il valore del bit e la loro direzione è la seguente:

$$bit\,0 \rightarrow \frac{\hat{\mathbf{i}}'l}{\sqrt{2}} + \frac{\hat{\mathbf{j}}'l}{\sqrt{2}}$$
$$bit\,1 \rightarrow \frac{\hat{\mathbf{j}}'l}{\sqrt{2}} - \frac{\hat{\mathbf{i}}'l}{\sqrt{2}}$$

$\hat{\mathbf{i}}$' e $\hat{\mathbf{j}}$'

rappresentano i vettori direzionali dell'asse x' e dell'asse y' solidali con il signor Ferrari.
Se consideriamo che la velocità della matita è nulla rispetto a Ferrari che la sta guardando, allora lo stato della matita è ben definito in senso classico.
Lo stato della matita è lo stesso sia che lo si guardi dal sistema solidale con Ferrari sia che lo si guardi dal sistema solidale con il terreno, cioè quello in cui l'asse y è la verticale rispetto al suolo.

Quindi lo stesso stato, per esempio bit 0, può essere scritto sia come sovrapposizione di stati

$$\left(\frac{\hat{\mathbf{i}}'_l}{\sqrt{2}} + \frac{\hat{\mathbf{j}}'_l}{\sqrt{2}} \right)$$

che some singolo stato

$$\left(\hat{\mathbf{i}}_l \right)$$

Quando il signor Ferrari esegue una misura dei *bit-matita* orizzontali, per esempio usando il goniometro, la misura darà sempre un risultato certo, cioè 0 gradi nel sistema solidale con il terreno e 45 gradi nel sistema solidale con esso, questo indipendentemente dal sistema scelto per rappresentare lo stato.
In pratica, gli stati della meccanica classica possono essere espressi come combinazioni di altri stati, ma questo non influisce sulle misure delle variabili che caratterizzano lo stato.

3.6 Idea quantistica

L'idea quantistica di sovrapposizione di stati non è così dissimile da quella classica. La introduciamo usando l'analogo quantistico dell'esempio delle matite: i fotoni del signor Fabbri. Rappresentiamo il vettore di polarizzazione del fotone con il simbolo $\hat{\mathbf{e}}$. Ricordiamo che il signor Magri invia fotoni con polarizzazione $\hat{\mathbf{e}}$ verticale o orizzontale al signor Fabbri il cui sistema di polaroid è ruotato di 45 gradi. Nel sistema del signor Magri, la polarizzazione orizzontale può essere espressa come:

$\hat{\mathbf{e}}=\hat{\mathbf{i}}$

mentre quella verticale come:

ê=ĵ

La polarizzazione del fotone con **ê=î** inviato dal signor Magri, nel sistema del signor Fabbri è invece vista come:

$$\hat{\mathbf{e}} = \frac{1}{\sqrt{2}} (\hat{\mathbf{i}}' + \hat{\mathbf{j}}')$$

Dove **î'** e **ĵ'** sono le direzioni gli assi ruotati dei sui polaroid. L'analogia tra meccanica quantistica e meccanica classica però finisce qui.

L'aspetto imprevisto e **caratterizzante** della fisica quantistica è che la conoscenza dello stato del sistema fisico non implica la capacità di predire il risultato di una misura.
Supponiamo che il signor Magri prepari un fotone nello stato

ê=î

e che prima di inviarlo avverta il signor Fabbri dello stato di polarizzazione in cui giungerà il fotone.
Il signor Fabbri attenderà il fotone e sarà pronto a misurarne lo stato nel suo sistema inclinato rispetto a quello di Magri, ma il risultato della misura sarà selettivamente o

ê=î'

oppure

ê=ĵ'

perdendo traccia dell'una o dell'altra componente. Quindi, una volta eseguita la misura, lo stato *collassa* in una delle due direzioni **ê=î'** o **ê=ĵ'**, perdendo traccia dello stato di *sovrapposizione*

$$\frac{1}{\sqrt{2}} (\hat{\mathbf{i}}' + \hat{\mathbf{j}}')$$

Domanda n. 10 Se un fotone si trova nello stato di polarizzazione $\hat{\mathbf{e}} = \frac{1}{\sqrt{2}}(\hat{\mathbf{i}} + \hat{\mathbf{j}})$, quale valore si deve attendere da una misura della polarizzazione?

1. $\hat{\mathbf{e}} = \frac{1}{\sqrt{2}}(\hat{\mathbf{i}} + \hat{\mathbf{j}})$
2. **ê=î**
3. **ê=î** oppure **ê=ĵ** con uguale probabilità

3.7 Stato e probabilità

Dal paragrafo precedente possiamo trarre alcune considerazioni riguardo alla misura dello stato di polarizzazione dei fotoni che hanno validità generale nella teoria quantistica.

Rispetto ad un certo strumento di misura, lo stato di polarizzazione di un fotone può essere espresso come combinazione lineare delle due direzioni $\hat{\mathbf{i}}$ e $\hat{\mathbf{j}}$. Con combinazione lineare si intende un'espressione del tipo

$\alpha\hat{\mathbf{i}} + \beta\hat{\mathbf{j}}$

dove α e β sono numeri complessi, cioè identificati da una lunghezza detta modulo e un angolo, detto fase.
Se lo stato di polarizzazione $\hat{\mathbf{e}}$ coincide con una delle due direzioni, cioè:

ê=î

oppure

ê=ĵ

la misura risulterà rispettivamente e deterministicamente o $\hat{\mathbf{i}}$ oppure $\hat{\mathbf{j}}$.
Lo stato di un sistema non cambia se viene moltiplicato per un numero complesso α di modulo 1 (cioè lunghezza pari ad 1 e angolo qualsiasi):

$\hat{\mathbf{e}}=\hat{\mathbf{i}}=\alpha\hat{\mathbf{i}}$

oppure

$\hat{\mathbf{e}}=\hat{\mathbf{j}}=\alpha\hat{\mathbf{j}}$

Si dice quindi che lo stato $\hat{\mathbf{e}}$ è definito a meno di una costante moltiplicativa e questo significa che **non c'è differenza fisica**[1] tra lo stato $\hat{\mathbf{i}}$ e lo stato $\alpha\hat{\mathbf{i}}$ se il numero complesso α ha modulo 1.
Se lo stato del fotone è in una combinazione lineare degli stati di base $\hat{\mathbf{i}}$ e $\hat{\mathbf{j}}$, cioè

$\hat{\mathbf{e}} = \alpha\hat{\mathbf{i}} + \beta\hat{\mathbf{j}}$

il risultato della misura potrà essere *indeterminatamente* sia $\hat{\mathbf{i}}$ che $\hat{\mathbf{j}}$. La probabilità di ottenere una delle due misure non è uguale alla probabilità di ottenere l'altra: la probabilità di ottenere $\hat{\mathbf{i}}$ è pari ad $|\alpha|^2$ mentre quella di $\hat{\mathbf{j}}$ è pari a $|\beta|^2$.
Si noti che è il rapporto tra α che β che **ha significato fisico** in quanto determina la probabilità di ottenere lo stato $\hat{\mathbf{i}}$ o lo stato $\hat{\mathbf{j}}$ come risultato di una misura.

3.7.1 Base di uno stato

Vediamo ora una considerazione di cui è difficile sopravalutare l'importanza.

[1] Vedi *Qubits: principi fondamentali* degli stessi autori.

Se la somma $|\alpha|^2 + |\beta|^2$ è uguale ad 1 allora significa che indipendentemente da ogni altra variabile che può caratterizzare il sistema, la polarizzazione dovrà sempre essere o lungo **î**, o lungo **ĵ**. In questo senso diremo che la polarizzazione costituisce una **base** per lo stato di un fotone.

La teoria impostata fino qui fornisce gli strumenti di calcolo per determinare la probabilità di ottenere un certo risultato misurando **un singolo** fotone che si trovi nel generico stato **ê**. Come si generalizza questo risultato quando lo stato fisico è relativo non ad uno ma a due, o più, fotoni?
La generalizzazione a più fotoni, ma in generale a più corpi o particelle, del risultato presentato, richiede l'introduzione del calcolo tensoriale.

3.8 Spazi vettoriali e prodotti tensoriali

Questo paragrafo ha un taglio molto matematico ma, per comprendere completamente i capitoli centrali dedicati all'informatica quantistica, è bene afferrare a fondo sia i concetti che il formalismo che vengono qui introdotti. Comunque, se questo risultasse troppo ostico, non ci si scoraggi, perché il resto può essere compreso **a livello intuitivo**, anche senza rigore matematico.

3.8.1 Base di uno spazio vettoriale

I tre versori **î**, **ĵ** e **û** sono una base dello spazio vettoriale R^3 e questo significa che qualunque vettore **v** dello spazio può essere

scritto come:

$\mathbf{v} = \alpha\hat{\mathbf{i}} + \beta\hat{\mathbf{j}} + \gamma\hat{\mathbf{u}}$

Nota: normalmente i versori si indicano con le lettere **i**, **j** e **k** con accento circonflesso, ma in questo testo viene usata la lettera **u** al posto della **k** per motivi tecnici di stampa. Una forma alternativa ai versori $\hat{\mathbf{i}}$, $\hat{\mathbf{j}}$ e $\hat{\mathbf{u}}$ utilizzati per indicare la base di R^3, è l'utilizzo degli elementi $\boldsymbol{e}_1$, $\boldsymbol{e}_2$ e $\boldsymbol{e}_3$. Questa notazione ha il vantaggio di poter essere usata anche per spazi vettoriali di dimensione superiore ($R^4, R^5, \ldots$). Infatti gli assi dello spazio non sono indicati con lettere diverse tra loro, ma usando un indice sotto la lettera $\boldsymbol{e}$ che ha il vantaggio di poter essere esteso a piacere con l'aumentare della dimensione dello spazio vettoriale senza la necessità inserire nuove lettere dell'alfabeto. Gli elementi $\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3, \ldots, \boldsymbol{e}_n$ costituiscono una base per un generico spazio vettoriale V di dimensione n.

Domanda n. 11 Come si scrive un vettore **v** che giace nel piano yz?

1. $\mathbf{v} = \boldsymbol{e}_1 + \boldsymbol{e}_2 + \boldsymbol{e}_3$
2. $\mathbf{v} = \alpha\boldsymbol{e}_1 + \beta\boldsymbol{e}_2$
3. $\mathbf{v} = \alpha\boldsymbol{e}_2 + \beta\boldsymbol{e}_3$

3.9 Base duale di uno spazio vettoriale

In questo paragrafo viene introdotto il concetto di elemento duale e di spazio vettoriale duale. Questo è un concetto nuovo e

normalmente estraneo a chi non ha condotto degli studi specifici di algebra. La cosa che potrebbe stupire è che gli elementi duali sono definiti in base alla loro azione sugli elementi $\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3, \ldots, \boldsymbol{e}_n$ anziché specificandone la loro natura. Questo è simile a descrivere un martello dicendo che è ciò che pianta i chiodi anziché descriverlo come un utensile composto da un manico in legno e una testa in metallo.

Accanto agli elementi $\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3, \ldots, \boldsymbol{e}_n$ introduciamo gli elementi $\boldsymbol{e}_1^*, \boldsymbol{e}_2^*, \boldsymbol{e}_3^*, \ldots, \boldsymbol{e}_n^*$ detti elementi duali dei primi.
Tra questi due insiemi di elementi esiste una relazione espressa come segue:

$$\begin{aligned}
&\boldsymbol{e}_1^*(\boldsymbol{e}_1) = 1 \\
&\boldsymbol{e}_2^*(\boldsymbol{e}_2) = 1 \\
&\ldots \\
&\boldsymbol{e}_1^*(\boldsymbol{e}_2) = \boldsymbol{e}_1^*(\boldsymbol{e}_3) = \cdots = 0
\end{aligned}$$

In modo più sintetico possiamo scrivere:

$$\boldsymbol{e}_i^*(\boldsymbol{e}_j) = \delta_{ij}$$

dove il simbolo δ_{ij} è detto delta di Kronecker e vale 1 se i due indici sono uguali, 0 altrimenti.
Come gli elementi $\boldsymbol{e}$ formano la base dello spazio vettoriale V, anche gli elementi $\boldsymbol{e}^*$ formano la base di un spazio vettoriale che chiameremo **spazio duale** di V.

3.10 Prodotto tensoriale

Introduciamo ora una nuova operazione che indichiamo con il simbolo $\otimes$ e che chiamiamo prodotto tensoriale.
Gli operandi del prodotto tensoriale sono sia gli elementi base $\boldsymbol{e}$ che i duali $\boldsymbol{e}^*$.
Consideriamo ora un secondo spazio vettoriale W accanto a V e, per non fare confusione con la base di V, indichiamo i suoi elementi di base con la lettera $\boldsymbol{f}$ anziché la $\boldsymbol{e}$: $\boldsymbol{f}_1, \boldsymbol{f}_2, \boldsymbol{f}_3, \ldots, \boldsymbol{f}_n$.
Definiamo il prodotto tensoriale fra due elementi, uno di V e uno di W, come segue:

$$\boldsymbol{e}_i \otimes \boldsymbol{f}_j$$

e, quando non crei ambiguità, scriviamo:

$$\boldsymbol{e}_i \otimes \boldsymbol{f}_j = \boldsymbol{e}_i \boldsymbol{f}_j$$

Con il formalismo visto sopra definiamo il prodotto tensoriale tra gli spazi V e W come:

$$V \otimes W =$$

$$= \boldsymbol{e}_1\boldsymbol{f}_1 + \boldsymbol{e}_1\boldsymbol{f}_2 + \boldsymbol{e}_1\boldsymbol{f}_3 + \cdots + \boldsymbol{e}_n\boldsymbol{f}_1 + \boldsymbol{e}_n\boldsymbol{f}_2 + \cdots + \boldsymbol{e}_n\boldsymbol{f}_n$$

Familiarizzare con il prodotto tensoriale è fondamentale perché permette di comprendere a fondo il concetto di **entanglement** che è uno dei fondamenti dell'informatica quantistica.

3.11 Prodotto tensoriale tra due vettori

Se $\mathbf{v}$ e $\mathbf{w}$ sono due vettori, il primo appartenente allo spazio V e il secondo alla spazio W, che ora per semplicità assumiamo essere lo stesso spazio ($V = W$), abbiamo che il loro prodotto tensoriale è dato dalla seguente espressione:

$$\mathbf{v} \otimes \mathbf{w} =$$
$$= v_1 w_1 \boldsymbol{e}_1 \boldsymbol{f}_1 + v_1 w_2 \boldsymbol{e}_1 \boldsymbol{f}_2 + v_1 w_3 \boldsymbol{e}_1 \boldsymbol{f}_3 + \dots$$
$$\cdots + v_n w_1 \boldsymbol{e}_n \boldsymbol{f}_1 + v_n w_2 \boldsymbol{e}_n \boldsymbol{f}_2 + \cdots + v_n w_n \boldsymbol{e}_n \boldsymbol{f}_n$$

Dove:

$$\mathbf{v} = v_1 \boldsymbol{e}_1 + \cdots + v_n \boldsymbol{e}_n$$

e

$$\mathbf{w} = w_1 \boldsymbol{f}_1 + \cdots + w_n \boldsymbol{f}_n$$

Dopo questa definizione generale, vediamo un esempio del prodotto tensoriale tra due vettori di R^2.
Siano $\mathbf{v} = 3\boldsymbol{e}_1 + 5\boldsymbol{e}_2$ e $\mathbf{w} = 2\boldsymbol{f}_1 + 4\boldsymbol{f}_2$ due vettori di R^2, il loro prodotto tensoriale è il vettore:

$$\mathbf{v} \otimes \mathbf{w} =$$
$$= 3 \times 2 \boldsymbol{e}_1 \boldsymbol{f}_1 + 3 \times 4 \boldsymbol{e}_1 \boldsymbol{f}_2 + 5 \times 2 \boldsymbol{e}_2 \boldsymbol{f}_1 + 5 \times 4 \boldsymbol{e}_2 \boldsymbol{f}_2 =$$
$$= 6\boldsymbol{e}_1 \boldsymbol{f}_1 + 12\boldsymbol{e}_1 \boldsymbol{f}_2 + 10\boldsymbol{e}_2 \boldsymbol{f}_1 + 20\boldsymbol{e}_2 \boldsymbol{f}_2$$

Da questa espressione si vede che il prodotto tensoriale dei due vettori genera un nuovo vettore e che tale vettore è espresso rispetto ai seguenti elementi:

$$\boldsymbol{e}_1 \boldsymbol{f}_1, \boldsymbol{e}_1 \boldsymbol{f}_2, \boldsymbol{e}_2 \boldsymbol{f}_1, \boldsymbol{e}_2 \boldsymbol{f}_2$$

Usando tali elementi è possibile esprimere ogni elemento dello spazio $R^2 \otimes R^2$ e quindi essi sono la base dello spazio $R^2 \otimes$

R^2. Essendo essi in numero quattro, si dice che la dimensione di questo spazio è 4.

Abbiamo visto che un qubit è associato ad uno spazio di dimensione due, dove ognuno dei due assi rappresenta una direzione di polarizzazione. Per rappresentare due qubits è necessario considerare il prodotto tensoriale tra due spazi di dimensione due analogamente a quanto fatto qui ora.

3.11.1 Stato di 2 qubits

Come lo stato di un qubit è rappresentabile per mezzo di un vettore dello spazio V, lo stato di 2 qubits è rappresentabile con un vettore dello spazio tensoriale ottenuto dal prodotto di $V \otimes V$ che ha dimensione 4, cioè 2×2.

Questo aspetto della teoria quantistica può colpire perché nella teoria classica lo spazio per descrivere due punti materiali non è il prodotto tensoriale $R^3 \otimes R^3$, ma il prodotto cartesiano $R^3 \times R^3$. In un sistema classico, infatti, la posizione di un singolo punto materiale è descritta da un vettore di R^3 e la la posizione di due punti materiali è descritta da un vettore di R^6 che ha dimensione $3 + 3$ e non 3×3.

La differenza rispetto al classico non risulta eclatante nel caso di due soli qubits perché 2×2 è uguale a $2 + 2$, ma diventa evidente se si considera un sistema formato da un numero elevato n di corpi che viene descritto in uno spazio di dimensione $2 \times 2 \times \cdots \times 2$ per n volte, quindi 2^n.

Riassumendo: nella visione classica, n corpi sono descritti da un vettore dello spazio $R^3 \times R^3 \times \cdots \times R^3$ per n volte, quindi

uno spazio con $n \times 3$ dimensioni (o assi). Nella teoria quantistica, lo stato di un sistema di n qubits è un vettore dello spazio $V^2 \otimes V^2 \otimes \cdots \otimes V^2$, moltiplicati per n volte, che ha 2^n dimensioni.

La differenza nel calcolo delle dimensioni dello spazio usato per descrivere un sistema quantistico rispetto ad uno classico, quando si passa da una ad n particelle, si spiega nel ruolo diverso che gioca lo stato fisico nei due diversi *paradigmi teorici*. Infatti, come abbiamo visto, allo stato quantistico di un qubit è associato il calcolo della probabilità di ottenere un dato risultato in conseguenza di una certa misura e il prodotto tensoriale è l'operazione che permette di scalare da 1 a n qubits coerentemente con la statistica.
Per questo motivo lo stato di n qubits è descritto dal loro prodotto tensoriale.
Nel seguito tratteremo sempre esempi con al massimo 3 qubits.

Domanda n. 12 Lo spazio per descrivere gli stati di un sistema di 3 qubits ha:

1. 3
2. 6
3. 8

dimensioni?

3.12 Stati entangled

Come è stato scritto nell'introduzione, non è necessario conoscere la fisica quantistica per apprezzare e sviluppare l'informatica quantistica ma, per apprezzarne a pieno il dense coding, la quantum teleportation e il no cloning principle, è necessario saper cosa sia l' entanglement.
Una descrizione puramente *pittorica* sarebbe stata più semplice, ma poco utile ai fini informatici.
Gli sforzi matematici per arrivare fin qui saranno ora premiati infatti, arrivati a questo punto, si avrà piena consapevolezza del perché nella meccanica quantistica è stato introdotto l' entanglement e cosa esso rappresenti veramente.

Abbiamo appena visto come si presenta il prodotto tensoriale tra due vettori.
Ogni vettore ottenuto dal prodotto di due vettori appartenenti allo spazio V dà luogo ad un altro vettore nello spazio vettoriale $V \otimes V$ ma, bisogna fare attenzione, perché *non è detto* il contrario. Vediamo infatti che esistono elementi dello spazio $V \otimes V$ che non possono essere scritti come il prodotto tra due vettori $\mathbf{v}$ e $\mathbf{w}$ dello spazio V: in questo caso vedremo che gli stati associati a tali elemento si dicono **entangled**.
Per vedere un esempio, consideriamo la seguente espressione:

$$a\boldsymbol{e}_1\boldsymbol{f}_1 + b\boldsymbol{e}_2\boldsymbol{f}_2$$

e ci chiediamo se sia possibile trovare due vettori $\mathbf{v}$ e $\mathbf{w}$ tali che

$$\mathbf{v} \otimes \mathbf{w} = a\boldsymbol{e}_1\boldsymbol{f}_1 + b\boldsymbol{e}_2\boldsymbol{f}_2$$

Per rispondere sviluppiamo il prodotto del termine a sinistra ($\mathbf{v} \otimes \mathbf{w}$) ottenendo:

$$v_1 w_1 \boldsymbol{e}_1 \boldsymbol{f}_1 + v_1 w_2 \boldsymbol{e}_1 \boldsymbol{f}_2 + v_2 w_1 \boldsymbol{e}_2 \boldsymbol{f}_1 + v_2 w_2 \boldsymbol{e}_2 \boldsymbol{f}_2 =$$
$$= a \boldsymbol{e}_1 \boldsymbol{f}_1 + b \boldsymbol{e}_2 \boldsymbol{f}_2$$

Come si vede, la risposta è che non è possibile: infatti, perché le due espressioni risultino uguali, quella di sinistra non dovrebbe contenere i termini $\boldsymbol{e}_1 \boldsymbol{f}_2$ e $\boldsymbol{e}_2 \boldsymbol{f}_1$. Per eliminare $\boldsymbol{e}_1 \boldsymbol{f}_2$ deve essere nullo uno tra i coefficienti v_1 e w_2 e per eliminare $\boldsymbol{e}_2 \boldsymbol{f}_1$ deve essere nullo uno tra i coefficienti v_2 e w_1, in questo modo però sarebbero nulli almeno uno tra i prodotti $v_1 w_1$ e $v_2 w_2$ che sono i coefficienti di $\boldsymbol{e}_1 \boldsymbol{f}_1$ e $\boldsymbol{e}_2 \boldsymbol{f}_2$.
Abbiamo dimostrato che esistono stati di due qubits entangled, cioè che non corrispondono a nessuna descrizione di due qubits considerati singolarmente.
Questo risultato è assolutamente diverso da quello che accade per i bit classici, dove lo stato di un sistema di bits è sempre dato dal prodotto cartesiano degli stati dei singoli bit.

Lo stato di due qubits che non possa essere scritto come il prodotto dei singoli qubits è detto essere **entangled**.

Questo è un risultato che si applica a tutta la meccanica quantistica (elettroni, protoni, ecc...), non solo ai qubits, ma qui focalizziamo l'attenzione solo sugli aspetti legati all' entanglement tra qubits.

3.13 Prodotto tensoriale tra due duali

Quanto visto per i vettori si applica anche ai vettori **duali**.
Per $\mathbf{v}^*$ e $\mathbf{w}^*$ appartenenti agli spazi duali di V e W scriveremo:

$\mathbf{v}^* \otimes \mathbf{w}^* = = v_1 w_1 \boldsymbol{e}_1^* \boldsymbol{f}_1^* + v_1 w_2 \boldsymbol{e}_1^* \boldsymbol{f}_2^* + v_1 w_3 \boldsymbol{e}_1^* \boldsymbol{f}_3^* + \ldots \cdots + v_n w_1 \boldsymbol{e}_n^* \boldsymbol{f}_1^* + v_n w_2 \boldsymbol{e}_n^* \boldsymbol{f}_2^* + \cdots + v_n w_n \boldsymbol{e}_n^* \boldsymbol{f}_n^*$

3.14 Prodotto tensoriale tra vettori e duali

È importante anche definire il prodotto tra vettori e **duali**.
Per $\mathbf{v}$ e $\mathbf{w}^*$ appartenenti agli spazi V e duale di W scriveremo:

$\mathbf{v} \otimes \mathbf{w}^* = = v_1 w_1 \boldsymbol{e}_1 \boldsymbol{f}_1^* + v_1 w_2 \boldsymbol{e}_1 \boldsymbol{f}_2^* + v_1 w_3 \boldsymbol{e}_1 \boldsymbol{f}_3^* + \ldots \cdots + v_n w_1 \boldsymbol{e}_n \boldsymbol{f}_1^* + v_n w_2 \boldsymbol{e}_n \boldsymbol{f}_2^* + \cdots + v_n w_n \boldsymbol{e}_n \boldsymbol{f}_n^*$

3.15 Matrici e tensori

Nota bene: La trattazione completa ed esaustiva di questo argomento deve essere condotta su un testo dedicato come il Landau o il Dirac. Qui si cercherà di delineare il minimo formalismo necessario per comprendere gli aspetti dell'informatica quantistica, facendo comunque attenzione a non semplificare eccessivamente in modo da non introdurre concetti falsi o ambigui.

Nella meccanica quantistica gli stati fisici sono rappresentati dai vettori. Le trasformazioni fisiche che avvengono sugli

stati, per esempio l'azione di un campo elettrico su un elettrone, sono rappresentate come **operatori** che agiscono sugli stati fisici trasformandoli.
Dal punto di vista matematico gli operatori della fisica sono visti come dei tensori.

Consideriamo ancora un fotone descritto dal vettore di polarizzazione $\hat{\mathbf{e}} = \alpha \boldsymbol{e}_1 + \beta \boldsymbol{e}_2$. Il vettore può essere ruotato in senso orario di un angolo, per esempio di $\theta = \frac{\pi}{4}$ dall'operatore $\boldsymbol{R}$

$\hat{\mathbf{e}}' = \boldsymbol{R}\hat{\mathbf{e}}$

in modo da ottenere il vettore $\hat{\mathbf{e}}'$.

L'espressione usata per definire la rotazione è puramente formale, ma non dice nulla sulla relazione esistente tra $\hat{\mathbf{e}}$ ed $\hat{\mathbf{e}}'$. Per concretizzare la rotazione dobbiamo dare una forma ad $\boldsymbol{R}$. Per esempio consideriamo la seguente espressione per l'operatore $\boldsymbol{R}$:

$\boldsymbol{R} = \cos\theta \boldsymbol{e}_1 \boldsymbol{e}_1^* + sin\theta \boldsymbol{e}_1 \boldsymbol{e}_2^* - sin\theta \boldsymbol{e}_2 \boldsymbol{e}_1^* + \cos\theta \boldsymbol{e}_2 \boldsymbol{e}_2^*$

Ricordando l'azione tra i duali e i vettori, vediamo che applicando $\boldsymbol{R}$ ad $\hat{\mathbf{e}}$, si ottiene l'espressione per la rotazione del vettore $\hat{\mathbf{e}}$ di un angolo θ:

$(\cos\theta \boldsymbol{e}_1 \boldsymbol{e}_1^* + sin\theta \boldsymbol{e}_1 \boldsymbol{e}_2^* - sin\theta \boldsymbol{e}_2 \boldsymbol{e}_1^* + \cos\theta \boldsymbol{e}_2 \boldsymbol{e}_2^*)$
$(\alpha \boldsymbol{e}_1 + \beta \boldsymbol{e}_2) =$
$(\alpha \cos\theta + \beta \sin\theta)\, \boldsymbol{e}_1 + (\beta \cos\theta - \alpha \sin\theta)\, \boldsymbol{e}_2$

Ponendo $\alpha = 1$, $\beta = 0$ e $\theta = \frac{\pi}{4}$ l'espressione sopra si riduce a

$\hat{\mathbf{e}}' = \frac{1}{\sqrt{2}} (\boldsymbol{e}_1 - \boldsymbol{e}_2)$

come già visto nel paragrafo 2.3 in figura 1.

Abbiamo creato una forma concreta per un operatore di rotazione. Vediamo ora cosa succede all'operatore se agiamo a sinistra con un elemento del duale e a destra lo facciamo agire su un elemento dello spazio vettoriale:

$\boldsymbol{e}_1^* (\boldsymbol{R} (\boldsymbol{e}_1)) = \cos\theta$

Dal momento che si è agito con gli elementi $\boldsymbol{e}_1^*$ e $\boldsymbol{e}_1$ possiamo indicare il risultato ottenuto con R_{11}, cioè:

$R_{11} = \boldsymbol{e}_1^* (\boldsymbol{R} (\boldsymbol{e}_1)) = \cos\theta$

In maniera analoga, definiamo gli elementi R_{12}, R_{21} e R_{22}. Come si vede, questi possono essere pensati come gli elementi di una matrice R 2×2:

$$\mathbf{R} = \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix}$$

Questa relazione tra operatori e matrici è alla base del nome **meccanica delle matrici** data inizialmente da Heisenberg alla meccanica quantistica.
Nel resto del testo non sarà necessario eseguire direttamente questi conti, ma è importante sapere cosa si intende per rappresentazione di un operatore rispetto agli stati (o vettori) di base.

Domanda n. 13 Come si scrive la trasformazione che manda $\boldsymbol{e}_1$ in $\boldsymbol{e}_2$ e $\boldsymbol{e}_2$ in $\boldsymbol{e}_1$?

1. $K = \cos\theta \boldsymbol{e}_1 \boldsymbol{e}_1^* + sin\theta \boldsymbol{e}_1 \boldsymbol{e}_2^* + -sin\theta \boldsymbol{e}_2 \boldsymbol{e}_1^* + \cos\theta \boldsymbol{e}_2 \boldsymbol{e}_2^*$
2. $K = \boldsymbol{e}_1 \boldsymbol{e}_2^* + \boldsymbol{e}_2 \boldsymbol{e}_1^*$
3. $K = \boldsymbol{e}_1 \boldsymbol{e}_2^* - \boldsymbol{e}_2 \boldsymbol{e}_1^*$

3.16 Operatori unitari

Consideriamo ora un fotone il cui stato sia scritto come:

$\hat{\mathbf{e}} = \alpha\hat{\mathbf{i}} + \beta\hat{\mathbf{j}}$

o equivalentemente come

$\hat{\mathbf{e}} = \alpha\boldsymbol{e}_1 + \beta\boldsymbol{e}_2$

La probabilità che una misura della polarizzazione del fotone lungo $\hat{\mathbf{i}}$ o lungo $\hat{\mathbf{j}}$ è data dal modulo quadrato dei coefficienti α e β. La polarizzazione del fotone ammette solo due stati possibili quindi, in base a quanto visto prima, sappiamo che $|\alpha|^2 + |\beta|^2 = 1$.

Pensiamo ora ad un operatore $\boldsymbol{O}$ che agisca sul vettore $\hat{\mathbf{e}}$ trasformandolo come segue:

$$\hat{\mathbf{e}}' = \boldsymbol{O}\hat{\mathbf{e}} = \alpha'\boldsymbol{e}_1 + \beta'\boldsymbol{e}_2$$

Se l'operatore $\boldsymbol{O}$ rappresenta una trasformazione fisica come ad esempio una rotazione nello spazio, allora dobbiamo aspettarci che la trasformazione non abbia influito sulla natura fisica del fotone caratterizzato dai suoi due possibili stati di polarizzazione. Quindi, dopo la trasformazione, anche la somma dei moduli quadrati dei nuovi coefficienti α' e β' deve essere pari ad 1, perché questo significa che i due stati costituiscono ancora una base per il fotone.
Ci aspettiamo infatti che rimanga invariata la probabilità totale che la polarizzazione del fotone sia verticale od orizzontale. Quindi che l'operatore agisca conservando la probabilità.

Non tutti gli operatori agiscono in questo modo: gli operatori che godono di questa proprietà sono detti **operatori unitari**.

3.17 Trasformazione di base per uno stato

Abbiamo visto che le misure della polarizzazione ammettono due soli possibili risultati: verticale o orizzontale e che questo ci permette di assumere i vettori $\hat{\mathbf{i}}$ e $\hat{\mathbf{j}}$ come base per rappresentare ogni possibile stato di polarizzazione del fotone.
Da quanto visto nel paragrafo precedente, segue però che ogni altra base ottenuta dalla prima per mezzo di una **trasformazione unitaria** (cioè da un operatore unitario) conserva la probabilità e quindi può essere una base accettabile per descrivere lo stato di un fotone.

La base usata fin ora è detta **base standard**.
Agendo sui vettori della base standard con l'operatore R definito dalla matrice

$$\mathbf{R} = \begin{pmatrix} 1 & 1 \\ i & -i \end{pmatrix}$$

(si ricordi che i è l'unità immaginaria dei numeri complessi) si ottiene la base data dai vettori: $\hat{\mathbf{i}} + i\hat{\mathbf{j}}$ e $\hat{\mathbf{i}} - i\hat{\mathbf{j}}$ detta di **base di polarizzazione circolare**. Questa base è molto importante nell'ottica quantistica, ma in questo testo non ne faremo uso perché non serve ai fini introduttivi che ci siamo proposti.

3.17.1 Vettori bra e ket

Nell'introduzione informale, abbiamo introdotto i simboli $|0\rangle$ e $|1\rangle$ per rappresentare lo stato del fotone. Siamo ora in grado di collocarli correttamente nella teoria.

Nella meccanica quantistica i ket $|\psi\rangle$ sono vettori di uno spazio detto **spazio di Hilbert**, mentre i bra $\langle\psi|$ sono i duali dei vettori.

I ket della base standard per la polarizzazione del fotone sono $|0\rangle$ e $|1\rangle$.

Ora è possibile derivare la relazione tra la rappresentazione **quantistica** che usa bra e ket e quella **ondulatoria** della famosa equazione di Schrodinger che usa la funzione d'onda $\psi(x)$. Si supponga che un sistema fisico sia descritto dal ket $|a\rangle$. Lo stesso stato fisico potrebbe essere descritto per mezzo della funzione d'onda $\psi(x)$. La relazione tra i due è la seguente:

$\psi(x) = \langle a|x\rangle$

Spesso si usa anche scrivere $|\psi\rangle$ per indicare un dato stato fisico.

Nel seguito del testo non sarà fatto uso della funzione d'onda di Schrodinger che è stata introdotta per completezza.

Domanda n. 14 lo stato $\frac{1}{\sqrt{2}}(|11\rangle + |00\rangle)$ di un sistema composto da due qubits è uno stato entangled?

1. si
2. no

3.18 Osservabili fisiche

Abbiamo visto che, nella teoria quantistica, lo stato di un sistema fisico è rappresentato da un ket ed è naturale chiedersi come sono collocate, nella teoria, le grandezze fisiche come la quantità di moto, il momento angolare, il campo elettrico, quello magnetico, ecc.
In questo testo, queste grandezze non verranno usate direttamente, per cui non è necessario spingersi nella descrizione precisa di questo aspetto della teoria. È interessante però sapere che esse sono rappresentate come degli operatori detti **osservabili**.
Per essere una osservabile, un operatore deve essere hermitiano, cioè la matrice formata dai suoi elementi deve essere simmetrica e reale (senza componenti immaginari).

In questo capitolo si è visto cosa si intende per stato fisico di un sistema quantistico, come è possibile associare un qubit allo stato di un fotone e, cosa molto importante, si è visto cosa si intende per stato entangled. Si è inoltre sviluppato il formalismo necessario per poter leggere e interpretare le espressioni usate nei libri e negli articolo professionali. Con questo bagaglio faticosamente guadagnato, si può ora procedere con i prossimi capitoli per capire senza compromessi e semplificazioni i principi dell'informatica quantistica.

CAPITOLO 4

I calcoli dei computer consumano energia?

L'informatica quantistica è nata dallo sforzo di comprendere quali fossero le limitazioni della scienza informatica, dovute alle leggi della fisica.

Tra i primi ad interessarsi a questo problema ci sono stati i fisici Bennett, Fredkin, Toffoli e Feynman. All'inizio degli anni '80, Bennet suggerì che un computer potesse essere pensato come una macchina che trasforma **energia libera** in calore con lo scopo di compiere un *lavoro matematico*: una definizione alla quale si potrebbe non essere abituati, se non si guarda il computer dal punto di vista *meccanico* (Bennett, 1982).
D'altra parte i computer sono meccanismi costruiti usando componenti elettronici che dissipano calore sia per elaborare i dati sia per mantenerli nella memoria volatile. Quindi, da questa

prospettiva, la definizione di Bennet non è poi così peregrina come potrebbe sembrare a prima vista.
Alcuni decenni prima, von Neumann, aveva condotto i primi studi sugli elaboratori elettronici e sugli automi cellulari. Uno dei suoi argomenti di ricerca era stato proprio il calcolo dell'energia associata alla computazione e, con maggior precisione, si era interessato al calcolo dell'aumento dell'**entropia** durante la computazione.

4.1 Entropia

L'entropia S è una grandezza fisica molto importante nella descrizione dei sistemi fisici che sono coinvolti in processi di trasformazione, come ad esempio una pentola d'acqua che si sta scaldando fino all'ebollizione per cuocere la pasta, o un caminetto in cui la legna arde trasformando l'energia chimica in calore. Questa grandezza è altrettanto importante per descrivere in modo completo anche i sistemi che producono, trasformano e trasmettono *informazione*.

4.1.1 Definizione in fisica dell'entropia

Il concetto di entropia è stato introdotto da Carnot e Clausius per quantificare la tendenza naturale dei sistemi fisici ad evolvere verso una precisa direzione anziché un'altra. Per esempio lasciando cadere una goccia di inchiostro in un bicchiere d'acqua ci si aspetta che l'inchiostro si diffonda in modo disordinato (direzione attesa) anziché formando delle figure geometriche re-

golari (direzione alternativa).
Nei termini di temperatura (T) e calore (Q), l'entropia è definita come:

$$\mathbf{S} = \sum \frac{dQ_i}{T_i}$$

dove T_i è la temperatura assoluta e dQ_i è il calore scambiato in una trasformazione composta da un insieme discreto di passaggi, dove ogni passaggio è indicato con l'indice *i*.
Al fine di comprendere le motivazioni dell'informatica quantistica, è molto istruttiva la sua formulazione data da Boltzmann. Quest'ultima definizione di entropia è basata sulla statistica: molto rigorosa e sopratutto generalizzabile anche ad ambiti diversi dalla fisica.
L'entropia S può essere scritta in termini statistici come:

$$\mathbf{S} = k \sum p_j ln(\frac{1}{p_j})$$

dove p_j è la probabilità che il sistema si trovi nello stato $j - esimo$ e k è la costante di Boltzmann.

Box: entropia nel gioco dei dadi

Si consideri di avere a disposizione sei dadi e di disporli in modo ordinato in modo che la somma dei loro numeri dia 36. Perché si verifichi questa condizione è necessario che tutti i dadi rivolgano al cielo la faccia con il numero 6. Per ogni dado esiste una solo caso su sei per corrispondente alla faccia 6, quindi esiste un'unica disposizione di dadi che porta il sistema dei sei dadi a dare come somma 36.

Si consideri ora la situazione in cui la somma dei dadi dà come risultato il numero 21.
Usando la definizione di entropia data poco sopra, risulta immediato verificare che l'entropia per la disposizione dei sei dadi tutti risultanti nel numero 6 ha una bassa entropia, mentre la disposizione risultante nei dadi che danno come somma 21 ha un'alta entropia.
Questa analisi qualitativa ci è sufficiente per intuire che un sistema che si trovi in uno stato a bassa entropia, scivolerà più facilmente verso uno stato ad entropia più alto, piuttosto che viceversa.

Esperimento mentale

Per comprendere ancora più a fondo, si può fare un semplice esperimento mentale. Si immagini che i dadi siano contenuti in un vaso e che si trovino nello stato iniziale, ordinato, con tutte le facce rivolte verso il 6, in modo che la somma dia 36. Questa situazioni corrisponde ad un minimo di entropia.
Ora si scuota il vaso in modo che i dadi cambino il loro valore casualmente. Quando ogni dado si sarà fermato, si sommino i valori delle sei facce. Con ogni probabilità si troverà un valore diverso da 36. Se si ripete l'esperimento molte volte, la media dei valori dopo lo scuotimento del vaso, scivolerà un po' alla volta verso il 21.
Questo semplice esperimento ci fornisce una idea intuitiva di un principio molto importante: in natura, i sistemi tendono ad assumere configurazioni con entropia crescente.

Entropia e reversibilità

Sempre continuando con lo stesso esperimento, vediamo che il concetto di entropia è legato anche al concetto di irreversibilità di una trasformazione.

Scelta una faccia da uno a sei, la probabilità che lanciando sei dadi si ottengano sei facce uguali a quella scelta è indipendente dal numero scelto. Per esempio ottenere sei facce con il numero sei ha la stessa probabilità di ottenere sei facce con il cinque, il quattro ecc.

Immaginiamo ora di disporre i dadi allineati in modo che la faccia superiore sia il numero 6 e la faccia frontale sia il 4.
Dallo stato iniziale, in cui tutti i dadi hanno valore 6, incliniamo lentamente il vaso, in maniera che tutti i dadi rotolino nello stesso modo e si posizionino tutti sulla faccia del numero 4. È un esercizio difficile ma, almeno mentalmente, possiamo pensare di riuscire ad eseguirlo.
Ora che tutti i dadi sono sullo stato 4, possiamo eseguire il movimento inverso e riportare i dadi sul valore 6: abbiamo eseguito una trasformazione **reversibile**, caratterizzata dal non aver aumentato l'entropia totale del sistema.

L'argomentazione usata ha solo un valore intuitivo, la relazione esatta tra l'entropia e la reversibilità deve essere studiata su un testo di termodinamica dedicato, come ad esempio Termodinamica di Enrico Fermi.

Ai fini della comprensione del resto del testo, l'idea intuitiva che abbiamo delineato è sufficiente.

Domanda n. 15

Quante sono le possibili combinazioni di sei dadi che corrispondono alla somma 36?

1. Una sola
2. Sei

4.2 Termodinamica della computazione

In questo paragrafo vengono introdotti due concetti importantissimi nella *teoria dell'informazione* sia classica che quantistica: l'entropia dell'informazione e l'entropia algoritmica. Concetti necessari per capire l'idea di computazione reversibile ed irreversibile, la prima delle quali è alla base dell'informatica quantistica.

4.2.1 Entropia dell'informazione

Contemporaneo di von Neumann, il matematico Claude Shannon introdusse il concetto di **entropia dell'informazione** applicando la statistica al concetto di informazione (Shannon, 1948).
In modo molto diretto egli definì lo schema della **teoria dell'informazione** in cinque blocchi uniti da un canale, detto canale dell'informazione. Da un lato si ha la sorgente dell'informazione e il sistema di trasmissione, mentre dall'altro si trova il sistema di ricezione e la destinazione. Tra i due blocchi si trova la sorgente di *rumore* che può intervenire nella comunicazione

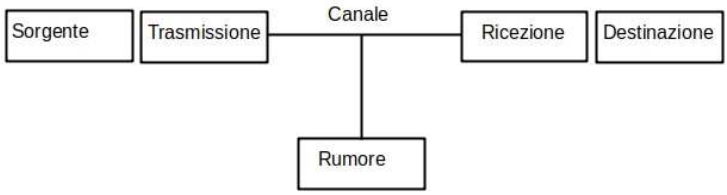


Figura 4.1: Schema del sistema di comunicazione proposto da C. Shannon.

alterando l'informazione.

L'informazione viene scambiata per mezzo di messaggi che sono semplicemente delle sequenze di simboli, per esempio i valori 0 e 1.

Partendo dall'idea che per ogni trasmissione l'informazione è trasportata da un messaggio e che ogni messaggio è una sequenza di simboli, egli definì p_i la frequenza con cui appare il simbolo i-esimo nel messaggio e mostrò che la funzione H definita come segue:

$$\mathbf{H} = \sum p_i log_2(\frac{1}{p_i}) = -\sum p_i log_2(p_i)$$

rappresentava l'**incertezza** contenuta nel messaggio. (Nota che la scelta della base 2 per il logaritmo, non è vincolante nella teoria.)

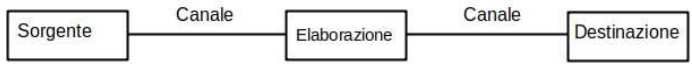


Figura 4.2: Sistema di comunicazione con un elemento di elaborazione dell'informazione tra sorgente e destinatario.

4.2.2 Un'applicazione della teoria dell'informazione di Shannon

La teoria dell'informazione e il concetto di **incertezza** o **arbitrarietà** contenuta in un messaggio, è piuttosto complessa e merita uno studio dedicato. Ai fini della comprensione di ciò che segue, dobbiamo però notare quanto la definizione di H ricalchi quella data per l'entropia S.
Consideriamo un circuito logico con due variabili di ingresso a, b e due di uscita a', b' composto da una porta AND e una OR definito dalla seguente tabella di verità:

a	b	a'	b'
1	1	1	1
1	0	0	1
0	1	0	1
0	0	0	0

Consideriamo ora una versione modificata dello schema a cinque blocchi visto sopra, in cui tra **sorgente** e **destinazione** introduciamo un blocco di **elaborazione**.

Per inquadrare la situazione secondo la teoria appena illu-

strata, dobbiamo considerare come simboli della trasmissione non i singoli bit, ma le coppie di bit. Quindi i simboli possibili per ogni messaggio trasmesso dalla sorgente sono:

$\{(1,1),(1,0),(0,1),(0,0)\}$

Nel caso in cui la sorgente abbia arbitrio totale sui simboli da inviare, essi hanno tutti probabilità $p = \frac{1}{4}$, quindi, usando la formulazione vista sopra per l'entropia, calcoliamo che l'**incertezza** dei messaggi trasmessi è 2 .
I possibili messaggi ricevuti, sono invece quelli derivabili dalle due colonne a', b' della tabella e, come si nota, si riducono a soli tre distinti tra loro:

$\{(1,1),(0,1),(0,1),(0,0)\}$

Il risultato dell'elaborazione è quindi quello di aver ridotto il numero di simboli possibili che possono essere ricevuti e di aver aumentato la probabilità (o frequenza) di uno di essi, cioè la coppia $(0,1)$.

È naturale chiedersi se l'elaborazione abbia avuto un effetto sulla quantità H definita poco prima.
Se calcoliamo H tenendo conto di quanto appena visto, vediamo che essa scende dal valore 2 al valore $\frac{3}{2}$. Inoltre notiamo un'altra cosa: alla variazione della funzione H è associato ad un altro concetto importante: il messaggio elaborato è **irreversibile**, cioè da esso non è più possibile risalire al messaggio trasmesso, perché per il simbolo ricevuto $(1,0)$ esistono due diversi possibili simboli trasmessi.

La funzione H è quindi un analogo dell'entropia statistica

vista sopra, e per questa analogia, pare che von Neumann abbia suggerito a Shannon di chiamarla entropia dell'informazione. Sembra inoltre che von Neumann abbia corroborato il suo suggerimento dicendo che visto che in pochi capivano il significato dell'entropia fisica, Shannon non avrebbe avuto critiche presentando il suo lavoro.

Originariamente Shannon aveva sviluppato e applicato la funzione H sopratutto ai problemi di trasmissione dell'informazione disturbati da sorgenti di rumore piuttosto che a trasmissioni trasformate da unità di elaborazione come nell'esempio presentato sopra. Tale argomento viene invece ripreso e sviluppato da Bennet che propone una sintesi del concetto di entropia algoritmica.

4.2.3 Entropia algoritmica

Vediamo ora di sviluppare il concetto di entropia dell'informazione di Shannon verso quello di entropia algoritmica seguendo la formalizzazione del problema come proposta da Bennet.
Egli definisce lo scopo di una computazione come quello di produrre una stringa di *output* x per mezzo dell'esecuzione di un programma.
Se questo dovesse sembrare strano, si pensi che qualsiasi risultato otteniamo da uno strumento informatico (digitale) è sempre una stringa di bit o una sua successiva *trasduzione* in un diverso segnale. Per esempio, se ascoltiamo della musica da un dispositivo informatico digitale, il suono che arriva alle nostre orecchie è il risultato della conversione di un segnale digitale

in uno elettrico analogico e successivamente in un segale acustico (onda di pressione).
Per continuare il ragionamento, definiamo la lunghezza di una stringa come il numero di bits da cui è composta. Definiamo poi l'**entropia algoritmica** come il numero minimo di bits per configurare un programma che produca l'output x.
Per afferrare il significato della grandezza appena introdotta, si provi ad immaginare di voler produrre delle stringhe casuali di bit. In questo caso, il programma non può sfruttare nessuna regola per produrre l'output x in quanto, se le stringhe sono casuali, la successione di 1 e 0 non ha seguito nessuna logica. Per configurare il programma serviranno quindi tanti bit quanti sono quelli che compongono la stringa x. Questo è il caso di **massima entropia**.
Adesso, si provi invece a considerare un programma che deve produrre la divisione intera per due di un dato input $a : x = \frac{a}{2}$. Dal punto di vista computazionale, per eseguire questo calcolo basta spostare tutti i bits dell'input a a destra di una unità, quindi, in linea di principio, il programma può essere configurato usando un unico bit di informazione, perché l'operazione di spostamento a destra del bit deve essere applicata a tutti i bit nello stesso modo. Questo è un caso di **minima entropia**.
Per chiarire ancora di più le idee, si consideri di voler ottenere che la stringa x sia ottenuta dalla divisione per 2 dell'input a nel caso a sia pari, mentre sia il risultato del prodotto per 2 sempre dell'input a nel caso a sia dispari. Semplificando molto il ragionamento, vediamo che sono necessari due bit per confi-

gurare il programma che esegue questa elaborazione. Il primo per stabilire se a è pari o dispari (in pratica controllando lo stato del bit 0) e il secondo per traslare a destra o a sinistra la stringa a.
Probabilmente, a qualcuno, non sfuggirà la stretta relazione esistente tra entropia algoritmica e la **complessità di Kolmogorov**.

Domanda n. 16

Quale affermazione tra quelle che seguono è falsa?

1. L'entropia dell'informazione è massima quando tutti i simboli hanno la stessa probabilità di presentarsi
2. L'entropia dell'informazione non è mai negativa
3. Le due precedenti affermazioni sono false

Entropia di un sistema

Prima di tirare le somme del ragionamento che abbiamo iniziato sull'entropia legata alla computazione, è necessario fare una piccola digressione sull'entropia di un sistema fisico.

La scienza che descrive la trasformazione del calore in lavoro e viceversa, è nota con il nome di termodinamica. La termodinamica si basa su tre principi, oppure può essere vista come una conseguenza statistica della teoria cinetica dei gas.
Nella termodinamica vi sono grandezze macroscopiche, come la temperatura, che sono il risultato del valore medio di grandezze microscopiche come l'energia cinetica delle particelle costituenti il gas. Tali grandezze macroscopiche hanno quindi una

controparte microscopica.
L'**entropia** invece, è una grandezza termodinamica macroscopica, presente sia nella descrizione classica che in quella statistica, che non ha una controparte microscopica. Non esiste quindi l'entropia di una particella e neanche una grandezza fisica della particella la cui media fornisca il valore dell'entropia del sistema.

Ora che abbiamo apprezzato questa caratteristica dei sistemi fisici, torniamo a considerare l'**entropia algoritmica** e vediamo che questa è l'analogo *microscopico* della entropia termodinamica *macroscopica*. In pratica si può calcolare l'entropia termodinamica se è nota l'entropia algoritmica di un sistema. Consideriamo un sistema termodinamico il cui comportamento macroscopico sia descrivibile in *modo conciso* (Bennet), cioè attraverso delle equazioni del moto macroscopiche, quindi senza bisogno di seguire le componenti del sistema ad una ad una. A questo proposito, Bennet, dice esplicitamente:

A macrostate is concisely describable, if it is determined by equations of motion and boundary conditions, describible in small number of bits...

Per un tale sistema, Bennet afferma:

...its statistical entropy is nearly equal to the ensamble average of microstates' algorithmic entropy

In altri termini, Bennet ci dice che per un sistema non del tutto *casuale*, quindi che abbia un certo grado di macro-determinismo, il grado di *casualità* è dovuto alla **entropia algoritmica** delle singole componenti del sistema stesso.

Questo implica che per trasformare la quantità $kT\ln(2)$ di **calore** in **lavoro** è necessario aumentare di un bit l'**entropia algoritmica** di un sistema fisico, e corrispondentemente, per diminuire di un bit l'entropia del sistema è necessario fornire un lavoro pari a $kT\ln(2)$.

Bennet sottolinea che questo è un risultato ben noto in termodinamica, ma che non era mai stato enunciato nei termini dell'entropia algoritmica, che appunto è una proprietà microscopica dei costituenti di un sistema.

Il risultato finale dell'argomentazione di Bennet, è che per modificare il valore di un bit è necessario spendere almeno un energia pari $kT\ln(2)$ che viene persa in calore.

I computer digitali sono costituiti da porte logiche che elaborano dati in forma binaria. Durante la computazione, ogni porta modifica il valore del proprio bit di output assegnandogli uno tra i valori 0 e 1, e lo fa indipendentemente dal valore del bit stesso. Quindi, per ogni bit assegnato durante una computazione, si ha una **riduzione** di $\ln(2)$ unità di entropia, che corrisponde alla produzione di calore pari a $kT\ln(2)$ alla temperatura assoluta T.

4.3 Termodinamica della computazione reversibile

La valutazione presentata nel paragrafo precedente dell'energia *sprecata* ad ogni assegnamento di un bit si basa sull'assunzio-

ne tacita che i processi elettronici usati per scrivere i bit sull'output delle porte logiche siano processi **irreversibili**.
Tra il 1980 e il 1982, furono elaborati due modelli di calcolo alternativi alla computazione ordinaria basati su principi di **calcolo reversibile**: il computer balistico e il computer browniano, che mostrarono esempi concreti, ma non pratici, di computazione reversibile. Nel prossimo paragrafo viene brevemente illustrato solo il computer balistico, il cui principio di funzionamento è più vicino alla meccanica che regola la computazione quantistica, mentre l'approfondimento del computer browninano viene lasciato all'iniziativa del lettore perché, per quanto assolutamente meritevole di approfondimento, ci porterebbe in una direzione diversa da quella voluta.

4.3.1 Il computer balistico reversibile

Il computer balistico fu presentato da Fredkin e Toffoli che illustrarono un *esperimento concettuale* per un sistema di calcolo che non dissipa energia, cioè produce *computazione* senza aumentare l'entropia totale del sistema.
Il computer balistico prevede un sistema di input che presenta un numero n di aperture circolari (porte) ed un sistema di output analogo con lo stesso numero di porte.
I bits del sistema sono delle sfere metalliche. Per presentare una stringa di input si preparano un numero m di sfere corrispondenti agli m bit di valore 1 nella stringa di input.
Le sfere vengono inserite con una certa velocità all'interno del computer, dove si trovano solo delle barriere metalliche contro

le quali le sfere possono rimbalzare variando (teoricamente) solo la direzione della velocità ma non l'intensità.
Dopo un certo numero di collisioni all'interno dell'elaboratore, le sfere emergono dalle porte di output con una certe disposizione. Le porte da cui emergono le sfere corrispondono ai bits 1 dell'output mentre quelle da cui non emerge alcuna sfera rappresentano i bits 0.
In condizioni ideali, sia gli urti tra le sfere stesse che quelli tra le sfere e le barriere, sono elastici e quindi non trasformano energia cinetica in calore. Quindi, questo tipo di computazione è reversibile.

4.3.2 Componenti circuitali per la computazione reversibile

In questo paragrafo vediamo che in linea di principio è possibile costruire un **computer reversibile** basato sul concetto comune di porte logiche.
Il computer balistico visto prima, infatti, è senz'altro affascinante, ma difficilmente può rivelarsi una soluzione pratica e realistica per eseguire computazioni di interesse pratico, per cui vedremo che è possibile costruire una macchina reversibile partendo dai componenti che sono usati concretamente nella realizzazione di normali computer *irreversibili*.

Ogni circuito logico può essere implementato anche usando due sole porte logiche, ad esempio il NOT e la AND.

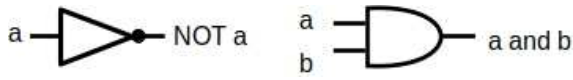


Figura 4.3: Simboli usati per le porte logiche NOT ed AND.

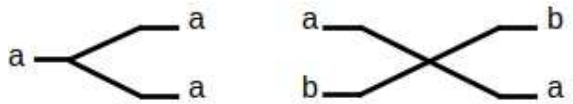


Figura 4.4: Simboli del FANOUT e dell'EXCHANGE.

Oltre a queste due porte, è di utilità pratica introdurre anche gli elementi FANOUT ed EXCHANGE (vedi figura sotto) che si rivelano indispensabili nella progettazione pratica dei circuiti, dove le variabili di input e output devono essere realizzate concretamente come cavi o linee di conduzione elettrica.

Partendo da questi quattro elementi circuitali vediamo come realizzarne le *controparti* reversibili e quindi come collegarle tra loro per realizzare dei circuiti logici. La proprietà che accomuna tutte le porte logiche che sono presentate di seguito è che il numero di variabili (indipendenti) di input è uguale al numero di variabili (dipendenti) di output. Senza questa condizione non è possibile ottenere una computazione reversibile.

4.3.3 Porta NOT reversibile

La porta NOT produce in output la negazione del segnale di input. Se un bit a viene presentato al suo ingresso, il bit NOT a

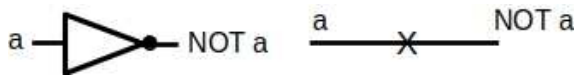


Figura 4.5: Simbolo classico e quantistico della porta NOT.

viene prodotto in uscita della porta. La porta NOT viene classicamente rappresentata con un triangolo, ma, seguendo la tradizione di Feynman, per evidenziare la natura *simmetrica* della porta NOT, la rappresentiamo con una X sopra un filo(cavo) conduttore con il quale si indica la linea di trasmissione dei bit.

4.3.4 Porta CONTROLLED NOT

Dopo il NOT ci si poteva aspettare di continuare con la porta reversibile AND, ma per farlo c'è bisogno prima di un nuovo elemento circuitale che diverrà molto importante e comune nel proseguimento dello studio della computazione quantistica: il **CONTROLLED NOT** (CNOT). Questa è una porta reversibile e quindi presenta lo stesso numero di ingressi e di uscite che indichiamo rispettivamente con le coppie a, b e a', b'.
La tabella di verità delle variabili è rappresentata qui sotto:

a	b	a'	b'
1	1	1	0
1	0	1	1
0	1	0	1
0	0	0	0

Il principio del CNOT è il seguente: il bit a ha il ruolo di *controllore*, mentre il bit b è il vero input. Se il bit a vale 0, allora il bit b viene trasmesso lungo la linea così com'è, quindi $b' = b$. Se invece il il bit a vale 1, allora il bit b viene negato. Il

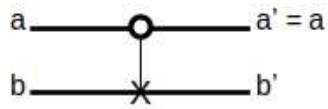


Figura 4.6: Simbolo della porta Controlled NOT (CNOT).

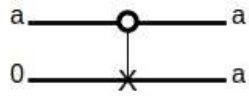


Figura 4.7: FAN OUT realizzato con una porta CNOT.

CNOT è rappresentato con il simbolo seguente:

4.3.5 FAN OUT

La porta CNOT può essere usata per produrre il FAN OUT (sdoppiamento di un segnale di ingresso). Se si pone il bit di ingresso $b = 0$, dalla tavola di verità della CNOT, si vede che $b' = a$ e $a' = a$, come mostrato nella figura seguente:

4.3.6 EXCHANGE

La porta CONTROLLED NOT può essere usata per produrre l'EXCHANGE (incrocio o scambio di segnali).
Lo scambio dei bit si ottiene ponendo tre CONTROLLED NOT in serie tra l'oro, in modo che la linea su cui è presente il bit di controllo risulti alternata tra un CNOT e il successivo. In termini pratici, l'EXCHANGE si ottiene con tre CNOT in serie,

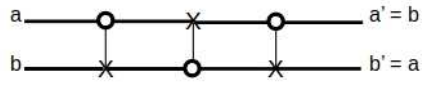


Figura 4.8: EXCHANGE realizzato con porte CNOT.

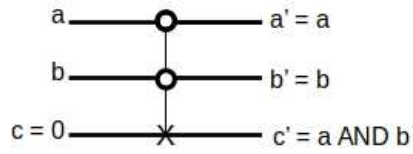


Figura 4.9: Circuito AND reversibile realizzato con due CNOT.

di cui quelle centrale ha il bit di controllo invertito rispetto ai due CNOT laterali, come mostrato nella figura seguente:

4.3.7 Porta AND reversibile

La porta AND **reversibile** può essere costruita usando due elementi CNOT in "parallelo" tra loro. L'elemento circuitale che si ottiene è mostrato nella prossima figura ed è nominato CONTROLLED CONTROLLED NOT (CCNOT).
Il funzionamento della CCNOT è il seguente: se entrambi i bit di controllo a e b sono settati ad 1, allora il bit c di ingresso viene negato, in modo da avere c' = NOT c, altrimenti il bit c viene trasmesso inalterato, quindi $c' = c$.
La porta AND, si ottiene quindi impostando a 0 il bit c.

La porta AND implementata come una CCNOT è reversibile, infatti, guardando la sua tabella di verità:

a	b	c	a'	b'	c'
1	1	0	1	1	1
1	0	0	1	0	0
0	1	0	0	1	0
0	0	0	0	0	0

risulta evidente che ad ogni terna di valori di output (sulla destra) corrisponde una sola terna di valori di input (a sinistra) e quindi dall'output è sempre possibile risalire all'input.

Domanda n. 17

Perché la porta AND *classica* è considerata una porta irreversibile?

1. Perché a fronte di 4 diverse combinazioni di ingresso può presentare solo due combinazioni diverse di uscita
2. Perché non è quantistica
3. Perché rappresenta una operazione booleana

4.3.8 Circuiti reversibili

Nei paragrafi precedenti sono stati introdotti dei nuovi elementi circuitali per implementare dei circuiti logici. Questi nuovi elementi possono essere usati al posto delle porte logiche e producono dei circuiti reversibili.

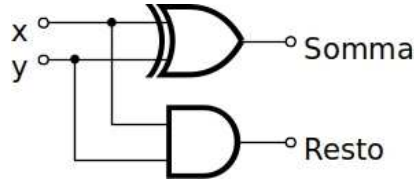


Figura 4.10: Circuito half-adder irreversibile, realizzato con porte logiche. Immagine tratta da Introduzione alle reti neurali degli stessi autori.

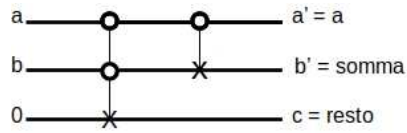


Figura 4.11: Circuito half-adder reversibile realizzato con CNOT.

4.3.9 Circuito half-adder

L'addizione tra due bit (x e y) vale 0 con resto di 1 se entrambi i bit sono 1, 1 con il resto di 0 se solo uno dei due bit vale 1, ed infine vale 0 con il resto di 0 se entrambi valgono 0.
L'implementazione classica e **irreversibile** di un circuito *half-adder* prevede ad esempio l'uso di una porta AND ed una porta XOR

La sua versione reversibile può essere realizzata usando tre linee di ingresso, un CCNOT e un CNOT come segue:

La tabella di verità, costruita usando le regole viste prima:

a	b	c	a'	b'	c'
1	1	0	1	0	1
1	0	0	1	1	0
0	1	0	0	1	0
0	0	0	0	0	0

mostra chiaramente che il circuito realizzato:

- implementa correttamente la semi-somma binaria (*half-adder*)
- è reversibile, perché per ogni configurazione di output esiste una sola configurazione di input

4.3.10 Circuito full-adder

L'addizione completa tra due bit prevede che si tenga conto di un eventuale bit di riporto dalla somma del binario precedente (x, y, r).
L'implementazione classica e **irreversibile** di un circuito *full-adder* prevede ad esempio l'uso di due porta AND, due porte XOR ed una porta OR:

La sua versione reversibile può essere realizzata usando quattro linee di ingresso, due CCNOT e due CNOT come segue:

Per calcolare la tabella di verità si presti attenzione al fatto che la variabile *d'* viene modificata due volte lungo il suo canale.

a	b	c	d	a'	b'	c'	d'
1	1	1	0	1	0	1	1
1	1	0	0	1	0	0	1

1	0	1	0	1	1	0	1
1	0	0	0	1	1	1	0
0	1	1	0	0	1	0	1
0	1	0	0	0	1	1	0
0	0	1	0	0	0	1	0
0	0	0	0	0	0	0	0

I valori delle variabili c' e d' che corrispondono alla somma e al resto dell'addizione tra a e b mostrano che il circuito realizzato:

- implementa correttamente la somma binaria completa (*full-adder*)
- è reversibile, perché per ogni configurazione di output esiste una sola configurazione di input.

4.4 Spazzatura binaria

Il circuito *full-hadder* appena costruito produce due bits di *spazzatura*, infatti i due output a' e b' non fanno parte delle informazioni ricercate, in pratica sono inutili ai fini informativi.
Come fece notare Feynman, la *spazzatura* è un elemento imprescindibile nella computazione reversibile, ma è proprio ciò che serve per poter tornare indietro, cioè risalire alla configurazione dei bit prima della loro elaborazione. Comunque, i bit inutilizzati, come b' possono essere ulteriormente trasformati per produrre qualcosa di utile. Ad esempio, se nel circuito del

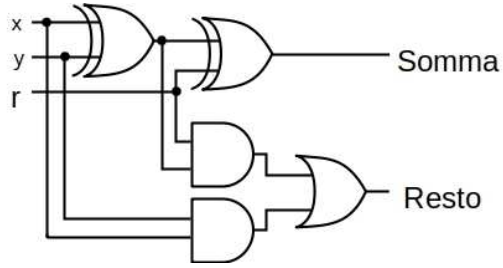


Figura 4.12: Circuito full-adder irreversibile..

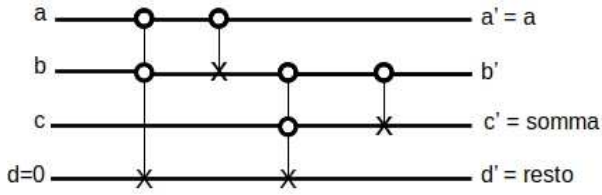


Figura 4.13: Circuito full-adder reversibile nella versione presentata da R. Feynman (1982).

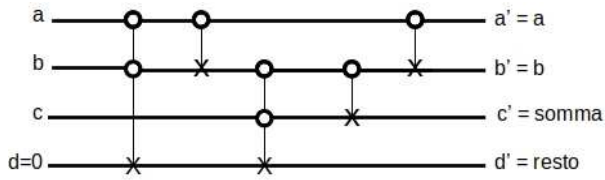


Figura 4.14: Circuito full-adder reversibile, con recupero del bit spazzatura.

full-adder viene aggiunto un CONTROLLED NOT sulla linea b' come mostrato di seguito:

le linee a e b risultano entrambe invariate dopo la computazione, e possono essere usate come ingressi per altri circuiti.

La proprietà appena vista si può generalizzare ad ogni circuito:

Per ogni circuito irreversibile con n bits di input e m di output è sempre possibile realizzare un circuito reversibile con $n+m$ bits di input dei quali m valorizzati a 0 e $n+m$ di output dei quali n valorizzati ai valori di input.

Domanda n. 18

Da chi fu presentato un circuito full-adder nel 1982?

1. Da Feynman
2. Da Shannon
3. Da Dirac

CAPITOLO 5

Gates quantistici

In questa sezione vediamo gli elementi alla base di un computer quantistico, cioè i sistemi fisici che seguono le regole imposte per i componenti **reversibili** visti nel capitolo precedente.

Le porte reversibili introdotte fin qui possono essere realizzate sfruttando le consuete tecnologie elettroniche. Quello che vedremo ora, invece, è come esse possano essere realizzate sfruttando le proprietà quantistiche della materia, proprietà che *spariscono* nel mondo macroscopico a cui siamo abituati, ma che sono la *vera legge* del mondo microscopico.

Nell'informatica quantistica, ci si riferisce a questi elementi circuitali indicandoli come **quantum gates**, in stretta analogia ai **logic gates** della computazione classica.
Nel seguito faremo uso di questo termine.

NOTA bene: nei capitoli introduttivi abbiamo visto come lo stato di polarizzazione di un fotone sia una buona osservabile (vedi paragrafo 3.5) per rappresentare il valore di un qubit. Anche le particelle *fermioniche* (cioè particelle a spin semi intero scoperte da Enrico Fermi) a *spin* uguale ad un mezzo possono essere usate per rappresentare i qubits e ancora altri sistemi quantomeccanici che posseggono solo due stati di base, ma in questo testo, considereremo sempre che il valore dei qubits sia associato allo stato di polarizzazione dei fotoni. I concetti illustrati sono indipendenti dalla reale tecnologia usata per ottenere e manipolare lo stato dei qubits.

5.1 Il NOT come gate quantistico

Consideriamo il qubit a che abbia valore 1 rispetto alla base standard. Possiamo scriverlo usando la notazione dei ket come segue:

$a = |1\rangle$

o equivalentemente come un vettore colonna:

$$\mathbf{a} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Vogliamo vedere come trasformare il valore del qubit $a = 0$ in $a' = 1$. Questa trasformazione non è un puro problema matematico, ma deve essere realizzabile a livello sperimentale se vogliamo capire come funziona nella realtà un computer quantistico.

La matematica ci guida nella comprensione dei fenomeni fisici e la fisica è la scienza della realtà, le leggi della fisica descrivono processi che avvengono in natura quindi se troviamo una trasformazione quantomeccanica che realizza l'operazione matematica voluta, troviamo anche l'operazione fisica da compiere per realizzarla.
Con questa premessa, vediamo che per trasformare il valore di a da 1 a 0, che è una operazione matematica, è necessario a livello fisico trasformare lo stato di un fotone da $|1\rangle$ ad $|0\rangle$ cambiandogli la polarizzazione agendo con uno strumento ottico che come vediamo qui di seguito dovrà compiere una trasformazione diversa da una semplice rotazione.

Dal punto di vista matematico formale, si tratta di trasformare il vettore

$$\mathbf{a} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

nel vettore

$$\mathbf{a}' = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

Detta trasformazione può essere ottenuta operando su a con la matrice

$$\mathbf{X} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

Si noti che il simbolo X usato per indicare la matrice è lo stesso usato nel simbolo del CNOT descritto precedentemente.

Dal punto di vista algebrico, la negazione di un qubit è quindi ottenuta moltiplicando la matrice X per il vettore a come segue:

$$\mathbf{a}' = \mathbf{X}\mathbf{a}$$

Dal punto di vista fisico invece la cosa è più complessa. Non sarà infatti sfuggito che la trasformazione X **non equivale** ad una rotazione nel piano. Le rotazioni nel piano xy attorno all'asse z, di un angolo θ misurato in senso orario, possono essere rappresentate (vedi paragrafo 3.3) con matrici 2×2 del tipo

$$\mathbf{R} = \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix}$$

che, come si deduce, non possono essere ridotte alla matrice X che ha i due elementi sulla diagonale secondaria dello stesso segno mentre per ogni angolo θ gli elementi sulla diagonale secondaria della matrice di rotazione R hanno sempre segno opposto (tranne quando valgono 0).

Per realizzare fisicamente la trasformazione X si usano degli strumenti ottici anche molto semplici come il *beam splitters* che consiste in due prismi di cristallo incollati tra loro.

Il circuito NOT per un singolo qubit è rappresentato graficamente come un quadrato (*box* in inglese) con una X.

5.1.1 Il NOT come trasformazione unitaria

Abbiamo visto che l'operazione NOT di un qubit equivale al prodotto della matrice X per il vettore rappresentante il suo stato

Figura 5.1: Simbolo dell'X gate.

di polarizzazione.
La matrice X è una trasformazione unitaria nel senso che si ottiene da un operatore unitario (paragrafo 3.3) e può essere rappresentata in forma tensoriale come segue::

$X = |0\rangle\langle 1| + |1\rangle\langle 0|$

La sua azione sui kets di base è data dalle trasformazioni:

$|1\rangle = (|0\rangle\langle 1| + |1\rangle\langle 0|)\,|0\rangle$

e

$|0\rangle = (|0\rangle\langle 1| + |1\rangle\langle 0|)\,|1\rangle$

La negazione di un qubit è quindi una operazione **quantistica** che agisce nello spazio degli stati del fotone, e questo significa che esiste una azione fisica **reversibile** che trasforma un qubit da 0 ad 1 e da 1 a 0, come anticipato nel paragrafo precedente.

Il risultato appena ottenuto ci dice che un'operazione logica come la negazione può essere ottenuta come una trasformazione quantistica reversibile che agisce su un singolo fotone. Quindi possiamo dire che:

per modificare il valore di un qubit è necessario trasformare lo stato del fotone che lo rappresenta.

A ben pensarci una affermazione equivalente è corretta anche per l'informatica classica, ma in quell'ambito la fisica è

più nascosta, mentre emergono problemi più ingegneristici che vengono trattati con più *senso comune* e meno formalismo fisico.

Prima di proseguire, notiamo un aspetto della computazione quantistica di straordinaria importanza (Rieffel, 2015 CAP 5), cioè che:

ogni calcolo può essere decomposto in trasformazioni che agiscono sul singolo qubit e in una trasformazione che agisce su due, come il CNOT.

in pratica, dopo aver definito la trasformazione NOT che agisce sul singolo qubit e la trasformazione CNOT che agisce su due, è possibile implementare qualsiasi operazione logica in termini quantistici.

Questo primo risultato evidenzia come la computazione automatica che attualmente è realizzata da circuiti elettronici irreversibili, può essere implementata nei termini di trasformazioni quantistiche reversibili. Aver dimostrato che è possibile replicare la computazione *classica* in termini quantistici, è ovviamente un passo necessario per dare valore scientifico all'informatica quantistica.

5.2 Matrici di Pauli

La matrice X che abbiamo introdotto nel paragrafo precedente è una delle tre matrici di *Pauli* definite come segue:

$$\sigma_{\mathbf{x}} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

$$\sigma_{\mathbf{y}} = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

$$\sigma_{\mathbf{z}} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Accanto alle matrici di Pauli vediamo le loro trasposte coniugate:

$$\sigma^*_{\mathbf{x}} = \sigma_{\mathbf{x}} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

$$\sigma^*_{\mathbf{y}} = \begin{pmatrix} 0 & i \\ -i & 0 \end{pmatrix}$$

$$\sigma^*_{\mathbf{z}} = \sigma_{\mathbf{z}} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Le matrici appena descritte sono matrici unitarie, cioè (ricordiamo) per loro vale la proprietà seguente:

$\sigma_x \sigma^*_x = \sigma_y \sigma^*_y = \sigma_z \sigma^*_z = I$

dove I è la consueta matrice identità definita come:

$$\mathbf{I} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Da ognuna delle matrici di Pauli, in quanto matrici unitarie, è possibile *teoricamente* ottenere un corrispondente **quantum**

gate.
Per passare dalla *teoria* alla *pratica*, bisogna verificare se esiste nella realtà un *setup sperimentale*, cioè un dispositivo concreto, che agisca concretamente sulla polarizzazione dei fotoni, così come l'operatore agisce sui kets.
A questa domanda fornirono una risposta di carattere generale Reck, Zeilinger, Bernsetin e Bertani che con una lettera pubblicata sulla rivista Physical Review Letters, presentarono un procedimento concreto per realizzare in laboratorio una trasformazione ottica da un sistema di input di N stati in un sistema di output di N stati usando solo *beam splitter*, *phase shifter* e *mirrors*. Testualmente scrivono: (Reck, 1994):

Per quel che ci risulta, questa è la prima volta che viene presentata una prova pratica che ad ogni operatore unitario discreto può essere associato un esperimento nella realtà.

Da questo, sappiamo che la *matematica* che stiamo scrivendo ha una corrispondenza nella realtà e abbiamo quindi la stessa tranquillità che ha un progettista elettronico nel creare funzioni logiche: sa che esiste un hardware che può implementarle.

5.2.1 Y e Z quantum gates

Il gate Y è costruito sulla matrice σ_y, ma è più pratico definirlo come $iY = \sigma_y$, cioè:

$$\mathbf{Y} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$$

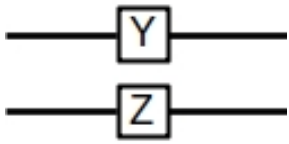


Figura 5.2: Rappresentazione grafica degli Y e Z gates.

Il gate Z corrisponde invece esattamente alla matrice σ_z:

$$\mathbf{Z} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

I due gate vengono rappresentati all'interno di un box quadrato analogamente al X gate.

In termini tensoriali, i gates possono essere scritti come:

$$\mathbf{Y} = -|1\rangle\langle 0| + |0\rangle\langle 1|$$

$$\mathbf{Z} = |0\rangle\langle 0| - |1\rangle\langle 1|$$

Il gate Z agisce sui kets nella **base standard** cambiandone il segno del rapporto tra a e b:

$(|0\rangle\langle 0| - |0\rangle\langle 1|)\,(a|0\rangle + b|1\rangle) =$

$= a|0\rangle - b|1\rangle$

L'azione del gate Y sullo stato $a|0\rangle + b|1\rangle$ è scritta come:

$(-|1\rangle\langle 0| + |0\rangle\langle 1|)\,(a|0\rangle + b|1\rangle) =$

$= -b|0\rangle + a|1\rangle$

e può essere vista come l'azione combinata del gate X e del gate Z, infatti si ha $Y = ZX$.

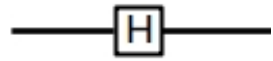


Figura 5.3: Simbolo dell'Hadamard gate.

5.3 Il gate H

Con il simbolo di una H dentro un box quadrato si indica la trasformazione di Hadamard, definita dalla matrice:

$$\mathbf{H} = \left(\; 1/\sqrt{2} \; \right) \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

o in forma tensoriale:

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

Per l'operatore H valgono le stesse considerazioni fatte per le matrici di Pauli quindi può essere realizzato concretamente anche utilizzando degli elementi di ottica lineare.

Vedremo il suo utilizzo, poco più avanti, per decodificare le informazioni inviate in modo: *dense coding*

5.4 Il gate CNOT

L'azione del CNOT può essere definita solo se si considera un sistema formato da due qubits. Quando questi sono espressi rispetto alla **base standard** allora, come definito nel capitolo precedente, la sua azione è di *negare* il secondo qubit se il primo è 1 o lasciarlo inalterato se il primo è 0.

Prima di analizzare questo gates è necessario formalizzare un insieme di kets di base per rappresentare gli stati di due qubits, ovviamente la formalizzazione che segue poggia su quanto visto nel paragrafo 3.3.

5.4.1 Base standard per due qubits

La base standard per un sistema formato da due qubits è data dal prodotto tensoriale delle due basi singole, quindi dai quattro ket:

$|0\rangle \otimes |0\rangle$, $|0\rangle \otimes |1\rangle$, $|1\rangle \otimes |0\rangle$, $|1\rangle \otimes |1\rangle$

che possono essere scritti più concisamente come segue:

$|00\rangle$, $|01\rangle$, $|10\rangle$, $|11\rangle$

In forma vettoriale, i quattro ket di base si rappresentano come segue:

$$|\mathbf{00}\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$|\mathbf{01}\rangle = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

$$|\mathbf{10}\rangle = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

$$|\mathbf{11}\rangle = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

quindi come quattro vettori colonna che risultano comodi quando si passa al calcolo matriciale.

5.4.2 Rappresentazione tensoriale e matriciale del gate CNOT

Ora che abbiamo definito un insieme di kets di base per gli stati di due qubits, possiamo vedere come si presenta in forma tensoriale il gate CNOT:

$CNOT = |00\rangle\langle 00| + |01\rangle\langle 01| + |11\rangle\langle 10| + |10\rangle\langle 11|$

e anche come si rappresenta in forma matriciale:

$$\mathbf{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

In entrambe le rappresentazioni è immediato verificare la correttezza dell'azione di trasformazione del CNOT sugli elementi della base:

$CNOT|00\rangle = |00\rangle$, $CNOT|01\rangle = |01\rangle$,
$CNOT|10\rangle = |11\rangle$, $CNOT|11\rangle = |10\rangle$

Infatti, considerando a titolo di esempio la trasformazione

$CNOT|10\rangle$

vediamo che essa può essere scritta come:

$(|00\rangle\langle 00| + |01\rangle\langle 01| + |11\rangle\langle 10| + |10\rangle\langle 11|)\,|10\rangle$

e valutando i quattro termini del prodotto si vede che tre sono nulli:

$|00\rangle\langle 00|10\rangle = 0$
$|01\rangle\langle 01|10\rangle = 0$
$|10\rangle\langle 11|10\rangle = 0$

e solo il termine

$|11\rangle\langle 10|10\rangle$

è diverso da 0 e dà come risultato esattamente $|11\rangle$ che rappresenta la negazione del secondo qubit dello stato di partenza $|10\rangle$.

È altrettanto immediato verificare che il CNOT non esegue il suo *servizio* se lo stato di input è una sovrapposizione degli stati di base.
Se consideriamo un generico ket dato dalla seguente espressione:

$|\psi\rangle = a|00\rangle + b|01\rangle + c|10\rangle + d|11\rangle$

si può verificare che l'azione del CNOT su di esso è la seguente:

$$(|00\rangle\langle 00| + |01\rangle\langle 01| + |11\rangle\langle 10| + |10\rangle\langle 11|)\,|\psi\rangle =$$
$$= a|00\rangle + b|01\rangle + d|10\rangle + c|11\rangle$$

che produce un nuovo ket che in generale **non** corrisponde alla negazione del secondo qubit. Quindi è importante capire che il gate CNOT si *comporta* come un NOT logico solo per qubit che si trovano in uno degli stati di base definiti prima, mentre se un qubit si trova in uno stato diverso dallo stato di base, il CNOT ne trasforma le componenti come mostrato qui sopra, ma non è possibile associare una funzione *logica* (AND, OR, NOT) alla trasformazione.

Per fissare bene le idee sull'azione del CNOT, consideriamo ad esempio il ket dato dalla sovrapposizione seguente:

$|\psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |10\rangle)$

Questo rappresenta un sistema di due fotoni dei quali uno è nello stato $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, e l'altro è nello stato $|0\rangle$. Lo stato $|\psi\rangle$, dei due fotoni presi insieme, non rappresenta nessuno dei quattro stati di base.

IL CNOT agisce sullo stato $|\psi\rangle$ trasformandolo in $|\psi'\rangle$ come segue:

$$|\psi'\rangle = CNOT\left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes |0\rangle\right) =$$
$$= \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

Nel risultato ottenuto ci sono due cose molto importanti da notare riguardo a :

- $|\psi'\rangle$ non rappresenta nessuno dei quattro stati di base del sistema
- $|\psi'\rangle$ è uno stato **entangled**

La prima considerazione ci rivela un risultato in realtà già atteso, abbiamo infatti definito l'effetto di CNOT solo sugli stati di base, ma non sugli stati dati da sovrapposizione di questi, quindi non ci sorprende che se il sistema dei due qubit si trova in uno stato di sovrapposizione che non rappresenta esattamente nessuno dei quattro stati in cui i qubits sono esattamente definiti a 0 o a 1, il risultato non sia quello voluto per il CNOT.

La seconda considerazione è che lo stato $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ ottenuto è uno stato di **entanglement** perché non può essere ottenuto in alcun modo come prodotto tensoriale dello stato di due fotoni distinti, infatti se fosse possibile ottenere $|\psi'\rangle$ dal prodotto tensoriale dello stato di due fotoni, questo dovrebbe essere scritto come segue:

$$(\alpha|0\rangle + \beta|1\rangle)(\gamma|0\rangle + \delta|1\rangle) =$$
$$= \tfrac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

quindi:

$$\alpha\gamma|0\rangle \otimes |0\rangle + \alpha\delta|0\rangle \otimes |1\rangle + \beta\gamma|1\rangle \otimes |0\rangle + \beta\delta|1\rangle \otimes |1\rangle =$$
$$= \tfrac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

Riscrivendo i ket di base si ha:

$$\alpha\gamma|00\rangle + \alpha\delta|01\rangle + \beta\gamma|10\rangle + \beta\delta|11\rangle =$$
$$= \tfrac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

dalla quale vediamo che affinché le due espressioni siano uguali devono scomparire i termini $|01\rangle$ e $|10\rangle$. Quindi devono essere posti uguali a 0 uno tra i coefficienti α e δ e uno tra β e γ. Così facendo però si annullano automaticamente anche i coefficienti dei ket $|00\rangle$ e $|11\rangle$ (come già dimostrato nel capitolo

3).

In conclusione abbiamo che lo stato $|\psi'\rangle$ ottenuto dalla trasformazione CNOT dello stato $|\psi\rangle$ non può essere visto come il prodotto di due stati, o in termini fisici possiamo dire che:

Lo stato entangled tra due fotoni non può essere considerato come lo stato di due fotoni separati e considerati insieme, ma è invece lo stato di un unico sistema che va considerato nella sua totalità

In questo paragrafo abbiamo visto una realtà pratica in cui si producono gli stati fisici entangled che avevamo presentato nel capitolo 3 solo come risultato matematico.
Nel capitolo 6, vedremo che tali stati sono alla base delle caratteristiche peculiari della computazione quantistica.

5.5 Altri gates a due qubits

Nel capitolo 4 abbiamo introdotto il simbolo grafico per rappresentare un CNOT all'interno di un circuito. Lo stesso simbolo è usato anche per rappresentare il *quantum gate* CNOT che abbiamo appena introdotto.

Il concetto di *gate controllato* può essere esteso anche ad altre trasformazioni oltre la X vista prima.
Si usa la lettera Q per indicare una generica trasformazione quantistica e un generico gate controllato è rappresentato con il simbolo seguente:

Sebbene non ne faremo ulteriore uso nel testo, è importante sapere cosa significa qualora lo si incontrasse altrove.

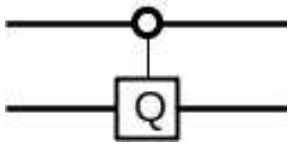


Figura 5.4: Simbolo del generico CQ gate.

Domanda n. 19

Le matrici di Pauli sono tali che moltiplicate per la propria trasposta coniugata risultano nella:

1. Matrice nulla
2. Matrice identità
3. Matrice quantistica

CAPITOLO 6

Computazione quantistica

Nei capitoli precedenti abbiamo visto che la computazione classica può essere *ripensata* in termini reversibili e che questo permette di creare una relazione uno a uno tra i componenti classici reversibili e delle trasformazioni quantistiche che agiscono a livello di sistemi atomici e subatomici.

Questo risultato di per sé sarebbe già sufficiente a giustificare a pieno l'interesse e la ricerca per la computazione quantistica, infatti basti pensare che mentre un qubit può essere realizzato anche usando un **singolo atomo** di materia, in un **grammo di silicio** ci sono circa 2×10^{22} atomi. Eppure non basta, c'è di più.

La vera ragione per cui la computazione quantistica sta emergendo in questi anni non è in sé la nanotecnologia a cui potrebbe portare (investigata giusto in questi ultimi anni), ma l'apertura a nuove frontiere della computazione non raggiungibile

con l'informatica classica. Problemi algoritmici che attualmente richiedono ore, giorni, settima o mesi con l'attuale tecnologia, possono teoricamente essere risolti su scale di tempi minimi usando la computazione quantistica.

In questo capitolo presentiamo finalmente le tre caratteristiche peculiari dell'informatica quantistica :

- il teorema di non clonazione dei qubit
- il *dense coding*
- il teletrasporto

che la distinguono rispetto alla classica.

6.1 Teorema di non clonazione

Sono stati presentati tutti gli elementi teorici che permettono di capire veramente il fascino e il valore di questo teorema senza doversi accontentare di una spiegazione qualitativa.

Il principio di non clonazione ci dice che:

Non è possibile copiare in modo affidabile lo stato di un sistema quantistico A in un altro sistema quantistico B se lo stato di A non è noto.

Abbiamo visto che le operazioni sui qubit vengono eseguite attraverso **trasformazioni unitarie**. Dimostriamo il teorema mostrando un caso avverso.

Supponiamo (**per assurdo**) che sia possibile copiare lo stato di un sistema sconosciuto in un secondo sistema, e che sia U

l'operatore unitario per eseguire tale trasformazione.
Sia $|a\rangle$ lo stato ignoto del sistema da clonare e $|0\rangle$ lo stato (noto o ignoto) del sistema in cui si desidera clonare $|a\rangle$, allora si deve avere

$$U\left(|a\rangle \otimes |0\rangle\right) = |a\rangle \otimes |a\rangle$$

Supponiamo ora che il ket $|b\rangle$ sia **ortogonale** ad $|a\rangle$ e che si voglia copiare un nuovo stato $|c\rangle = \frac{1}{\sqrt{2}}\left(|a\rangle + |b\rangle\right)$ Per la linearità degli operatori unitari, si dovrebbe avere:

$$U\left(|c\rangle \otimes |0\rangle\right) = \frac{1}{\sqrt{2}}\left(U|a\rangle \otimes |0\rangle + U|b\rangle \otimes |0\rangle\right) =$$
$$= \frac{1}{\sqrt{2}}\left(|a\rangle \otimes |a\rangle + |b\rangle \otimes |b\rangle\right)$$

ma lo stato ottenuto è diverso dal risultato che si attende dalla definizione data di clonazione, cioè:

$$U\left(|c\rangle \otimes |0\rangle\right) = |c\rangle \otimes |c\rangle =$$
$$= \frac{1}{\sqrt{2}}\left(|a\rangle \otimes |a\rangle + |a\rangle \otimes |b\rangle + |b\rangle \otimes |a\rangle + |b\rangle \otimes |b\rangle\right)$$

Da questa semplice dimostrazione, si vede che non può esistere una trasformazione unitaria che possa garantire la clonazione di uno stato quantistico.
Si noti che in certi casi potrebbe essere possibile, ma visto che sicuramente ci sono casi per i quali non è possibile, la clonazione sarebbe **non affidabile**.
È molto importante notare che:

Il teorema di non clonazione si applica solo a stati ignoti. Se si conosce lo stato di un sistema, è possibile preparare un altro sistema nello stesso stato quantistico.

Una nota per chi già ha conoscenza della meccanica quantistica: si faccia attenzione a non mal interpretare quest'ultima affermazione che potrebbe apparire in contraddizione con

il **principio di esclusione di Pauli**, infatti, anche nel caso di qubits fermionici, la copia di uno stato significa copiare lo stato di un sistema in un diverso sistema, e quindi non è in contraddizione con detto principio.

6.1.1 Conseguenze del teorema di non clonazione

A causa del teorema enunciato, nella computazione di qubits e nella loro trasmissione non è possibile applicare le tecniche di rivelazione e correzione degli errori usate nella computazione classica.
La computazione quantistica e la trasmissione di dati quantistici è soggetta a diverse sorgenti di errori: rumore, errori di trasformazione dovuti ai gates, errori di preparazione dei qubits e di misura degli stessi. Sarebbe difficile, per questo motivo, accettare la computazione quantistica se non esistesse una tecnica di correzione degli errori.
Per fortuna, questa è stata individuata e sviluppata a metà degli anni '90 e fa largo uso dell' entanglement. La sua descrizione e comprensione va però oltre gli obiettivi di questo testo, dove ci limitiamo a sottolineare l'importanza dell'argomento.

Il teorema di non clonazione ha ovviamente molta rilevanza dal punto di vista della sicurezza dei dati. Si faccia però attenzione al fatto che il teorema afferma che non sia possibile clonare dei dati con precisione, ma lascia uno spiraglio a chi volesse clonarli accettando un certo grado di imprecisione: cracker che si accontentano!

Comunque bisogna sempre stare attenti, perché la sicurezza informatica è un tema caldo e attuale anche con le tecnologie quantistiche e i rischi si possono nascondere dove meno uno se li aspetta, come nei *loopholes quantistici* (Jogenfors, 2017).

6.2 Dense coding

Il dense coding permette di comunicare due bits (non qubits) di informazione trasmettendo un solo qubit: da qui il nome *dense*. Questa possibilità non esiste nell'informatica e nella meccanica classica, ma non si deve pensare che sia una specie di *magia* quantistica, infatti sappiamo che:

> da ogni qubit è possibile estrarre un solo bit di informazione
> quindi, come avviene il dense coding?

La spiegazione c'è: il dense coding si basa sulla condivisione *pre-comunicazione* di uno stato **entangled**, dove *pre-comunicazione* significa che prima della trasmissione i due soggetti della comunicazione avevano in precedenza condiviso una risorsa comune analogamente a quello che potrebbe essere la condivisione di un disco su una local network.

BOX: Preparazione dello stato entangled lo stato entangled è la risorsa che i due condividono. Può sembrare un concetto astratto ma, in termini pratici, si tratta di realizzare un dispositivo che prepara un sistema fisico microscopico in uno stato quantistico entangled.
Per esempio è possibile preparare uno stato entangled dall'e-

missione, da parte di un atomo di calcio, di due fotoni emessi in *cascata*, cioè uno di seguito all'altro.
La cascata di due fotoni si ottiene eccitando i livelli atomici dell'atomo di calcio che successivamente torna nel suo stato fondamentale passando in successione prima dal livello 6^1S_0 al livello 4^1P_0 e successivamente al 4^1S_0.
Per eccitare i livelli atomici, il calcio viene semplicemente scaldato.
I fotoni che vengono emessi sulla stessa linea di volo ma con angoli opposti sono entangled ed hanno entrambi polarizzazione verticale oppure entrambi orizzontale e possono essere usati per il dense coding (Kocher, 1967).

Supponiamo che la risorsa condivisa sia lo stato entangled di due fotoni come descritto nel box sopra. Possiamo descriverla come:

$|\psi\rangle = \frac{1}{\sqrt{2}} (|0\rangle \otimes |0\rangle + |1\rangle \otimes |1\rangle)$

Si ricordi che la scrittura

$|0\rangle \otimes |0\rangle + |1\rangle \otimes |1\rangle$

equivale alla seguente

$|00\rangle + |11\rangle$

In questo contesto preferiamo la prima alla seconda perché evidenzia il fatto che il sistema fisico è condiviso tra due soggetti, quindi possiamo pensare il primo ket per il soggetto 1 ed il secondo per il soggetto 2. Oppure, ricordandoci dei signori Fabbri e Magri introdotti nei primi capitoli, possiamo riscrivere lo stato $|\psi\rangle$ specificando il ket di fabbri con la lettera maiuscola F e il ket di Magri con la M:

$|\psi\rangle = \frac{1}{\sqrt{2}} (|0_F\rangle \otimes |0_M\rangle + |1_F\rangle \otimes |1_M\rangle)$

6.2.1 Codifica in dense coding

Questa volta il signor Fabbri vuole comunicare una informazione al signor Magri. Il signor Fabbri infatti ha necessità di informare Magri circa la scelta da lui compiuta di uno dei quattro punti cardinali, questo perché Fabbri e Magri gestiscono un servizio di orientering.
Il signor Fabbri trasforma il proprio fotone della coppia entangled secondo il seguente schema:

- $Nord \rightarrow (I \otimes I)\,|\psi\rangle =$
  $= \frac{1}{\sqrt{2}} (|0_F\rangle \otimes |0_M\rangle + |1_F\rangle \otimes |1_M\rangle)$
- $Sud \rightarrow (X \otimes I)\,|\psi\rangle =$
  $= \frac{1}{\sqrt{2}} (|1_F\rangle \otimes |0_M\rangle + |0_F\rangle \otimes |1_M\rangle)$
- $Ovest \rightarrow (Z \otimes I)\,|\psi\rangle =$
  $= \frac{1}{\sqrt{2}} (|0_F\rangle \otimes |0_M\rangle - |1_F\rangle \otimes |1_M\rangle)$
- $Est \rightarrow (Y \otimes I)\,|\psi\rangle =$
  $= \frac{1}{\sqrt{2}} (-|1_F\rangle \otimes |0_M\rangle + |0_F\rangle \otimes |1_M\rangle)$

Le trasformazioni X, Z e I sono quelle viste nel capitolo precedente. Ricordiamo che il prodotto $U \otimes I$ indica che la trasformazione U è applicata al primo ket, mentre il secondo è lasciato inalterato, anche perché il secondo ket non si trova in possesso del signor Fabbri.
Quindi, se il signor Fabbri ha scelto l'**est** e vuole comunicarlo

a Magri, anzitutto trasforma il ket $|\psi\rangle$ nel ket

$$|\psi'\rangle = \frac{1}{\sqrt{2}}\left(-|1_F\rangle \otimes |0_M\rangle + |0_F\rangle \otimes |1_M\rangle\right)$$

agendo con l'operatore Y sul suo fotone, mentre il fotone di Magri (che non è a disposizione di Fabbri) rimane invariato perché trasformato per mezzo dell'identità I che non sortisce alcun effetto.

6.2.2 Trasmissione e ricezione

Dopo la trasformazione operata sul proprio fotone, il signor Fabbri lo trasmette al signor Magri il quale non misura subito lo stato del fotone per estrarre qualche informazione, ma trasforma la coppia di qubits ora in suo possesso con due trasformazioni in successione: prima con un CNOT e poi con l'operatore di Hadamard (H).
Dopo tali trasformazioni procede alla misura della coppia di fotoni entangled dalla quale estrarrà due bits di informazione in seguito alla trasmissione di un singolo qubit.
In base allo stato dei due fotoni egli risale alla direzione che il signor Fabbri voleva comunicare.

Riflettiamo su cosa è successo: i punti cardinali sono 4, quindi per selezionarne uno servono due bit classici:
$(i.e. 00, 01, 10, 11)$.
Quello che hanno fatto i signori Fabbri e Magri è stato comunicarsi una scelta, tra le quattro possibili, trasmettendosi un solo qubit!
Il *trucco* sta nell' entanglement. I qubits che i due hanno a

disposizione sono in effetti due (uno per uno) quini l'informazione **non** viene *creata dal nulla* piuttosto è un modo nuovo di gestirla che sfrutta una caratteristica della natura che non è evidente nella vita quotidiana.
Vediamo ora nei dettagli la decodifica del dense coding

6.2.3 Trasformazione e decodifica

Il signor Magri riceve il qubit descritto dal ket $|\psi'\rangle$ e lo trasforma in $|\psi''\rangle$ agendo prima con un CNOT gate e poi con un Hadamard gate come descritto dalla espressione seguente:

$$|\psi`'\rangle = H\,(CNOT|\psi'\rangle)$$

Di seguito, esegue la misura dello stato di polarizzazione di entrambi i qubits, giungendo ad una della quattro seguenti possibili combinazioni dei due qubits:

- $|0_F\rangle \otimes |0_M\rangle$
- $|0_F\rangle \otimes |1_M\rangle$
- $|1_F\rangle \otimes |0_M\rangle$
- $|1_F\rangle \otimes |1_M\rangle$

Il signor Magri, avendo eseguito il calcoli, sa che la prima combinazione corrisponde al nord, la seconda al sud, poi all'ovest e la quarta all'est.

Riassumendo, è possibile comunicare due bits classici di informazione, trasmettendo un singolo qubit.

BOX: calcoli espliciti Il calcolo dell'azione delle trasformazioni CNOT e H sullo stato $|\psi'\rangle$ si ottiene anzi tutto valutando l'azione della prima trasformazione su ognuno dei quattro possibili stati:

- $\frac{1}{\sqrt{2}}(|0_F\rangle \otimes |0_M\rangle + |1_F\rangle \otimes |1_M\rangle)$
- $\frac{1}{\sqrt{2}}(|1_F\rangle \otimes |0_M\rangle + |0_F\rangle \otimes |1_M\rangle)$
- $\frac{1}{\sqrt{2}}(|0_F\rangle \otimes |0_M\rangle - |1_F\rangle \otimes |1_M\rangle)$
- $\frac{1}{\sqrt{2}}(-|1_F\rangle \otimes |0_M\rangle + |0_F\rangle \otimes |1_M\rangle)$

dai quali si ottiene:

- $\frac{1}{\sqrt{2}}(|0_F\rangle \otimes |0_M\rangle + |1_F\rangle \otimes |0_M\rangle)$
- $\frac{1}{\sqrt{2}}(|1_F\rangle \otimes |1_M\rangle + |0_F\rangle \otimes |1_M\rangle)$
- $\frac{1}{\sqrt{2}}(|0_F\rangle \otimes |0_M\rangle - |1_F\rangle \otimes |0_M\rangle)$
- $\frac{1}{\sqrt{2}}(-|1_F\rangle \otimes |1_M\rangle + |0_F\rangle \otimes |1_M\rangle)$

Tali stati possono essere riscritti in modo da poter valutare più semplicemente l'azione dell'operatore $H \otimes I$:

- $\frac{1}{\sqrt{2}}(|0_F\rangle + |1_F\rangle) \otimes |0_M\rangle$
- $\frac{1}{\sqrt{2}}(|1_F\rangle + |0_F\rangle) \otimes |1_M\rangle$
- $\frac{1}{\sqrt{2}}(|0_F\rangle - |1_F\rangle) \otimes |0_M\rangle$
- $\frac{1}{\sqrt{2}}(-|1_F\rangle + |0_F\rangle) \otimes |1_M\rangle$

che rappresenta la nota trasformazione dalla **base di Hadamard** alla **base standard** e vice versa, quindi, essendo i ket tra parentesi i quattro ket di base della base di Hadamard, l'azione di $H \otimes I$ è quella di trasformarli nei ket di base della base standard. Applicando l'operatore segue immediatamente l'ultimo passaggio:

- $|0_F\rangle \otimes |0_M\rangle$
- $|0_F\rangle \otimes |1_M\rangle$
- $|1_F\rangle \otimes |0_M\rangle$
- $|1_F\rangle \otimes |1_M\rangle$

6.3 Teletrasporto

Il teletrasporto di uno stato quantico è forse la più suggestiva delle novità portate dalla meccanica e dalla computazione quantistica.
Bisogna però stare attenti e non lasciarsi prendere troppo dalla fantasia. Nel teletrasporto, ad essere teletrasportata è *un'informazione* e **non** la materia o l'energia in sé. In pratica quello che si ha è che l'informazione codificata in un qubit non è più accessibile in un luogo mentre diventa accessibile in un altro.

6.3.1 Setup sperimentale

Riprendiamo i signori Fabbri e Magri lasciati al paragrafo precedente. Il signor Fabbri ha un qubit rappresentato dal ket $|\phi\rangle$

che vuole trasmettere al signor Magri attraverso un canale di informazione classico, cioè inviandogli due bits.
Per fissare le idee, supponiamo che ϕ sia scritto nella base standard come segue: $|\phi\rangle = a|0\rangle + b|1\rangle$.
Come nel caso del dense coding, Fabbri e Magri condividono un sistema fisico composto da due fotoni entangled e descritto dallo stato $|\psi\rangle$.
Lo stato fisico totale del sistema è:

$$|\phi\rangle \otimes |\psi\rangle =$$
$$= (a|0_F\rangle + b|1_F\rangle) \frac{1}{\sqrt{2}} (|0_F\rangle \otimes |0_M\rangle + |1_F\rangle \otimes |1_M\rangle)$$

Di nuovo, l'indice F o M indica se il fotone è sotto il controllo di Fabbri o di Magri.
Il signor Fabbri controlla due fotoni, mentre Magri ne controlla uno. Essi inoltre si possono trasmettere due bits (classici) attraverso una certa rete di comunicazione non quantistica.

6.3.2 Preparazione e trasmissione dei dati

Il signor Fabbri trasforma i due fotoni che ha a disposizione agendo prima con un CNOT e poi con l'operatore di Hadamard. Più esplicitamente egli prima agisce con $CNOT \otimes I$ poi con $H \otimes I \otimes I$. Lo stato ottenuto è il seguente:

$$|\phi`\rangle \otimes |\psi'\rangle = \tag{6.1}$$

$$= \frac{1}{2}((a|0_M\rangle + b|1_M\rangle)|0_F\rangle \otimes |0_F\rangle + \tag{6.2}$$

$$+(a|1_M\rangle + b|0_M\rangle)|0_F\rangle \otimes |1_F\rangle + \tag{6.3}$$

$$+(a|0_M\rangle - b|1_M\rangle)|1_F\rangle \otimes |0_F\rangle + \tag{6.4}$$

$$+(a|1_M\rangle - b|0_M\rangle)|1_F\rangle \otimes |1_F\rangle) \tag{6.5}$$

Fabbri esegue la misura dei suoi due qubits e ottiene da ognuno dei due o $|0_F\rangle$ oppure $|1_F\rangle$ quindi in totale una tra le seguenti quattro possibili combinazioni:

- $|0_F\rangle \otimes |0_F\rangle$
- $|0_F\rangle \otimes |1_F\rangle$
- $|1_F\rangle \otimes |0_F\rangle$
- $|1_F\rangle \otimes |1_F\rangle$

Il signor fabbri copia la combinazione ottenuta in due bits classici che trasmetterà al signor Magri.

Anche il fotone sotto il controllo di Magri è rimasto coinvolto nella trasformazione in quanto entangled a quello di Fabbri. Lo stato in cui è stato trasformato il fotone di Magri è deducibile in base alla combinazione ottenuta dalla misura dei qubits operata da Fabbri:

- $|0_F\rangle \otimes |0_F\rangle \rightarrow (a|0_M\rangle + b|1_M\rangle)$
- $|0_F\rangle \otimes |1_F\rangle \rightarrow (a|1_M\rangle + b|0_M\rangle)$

- $|1_F\rangle \otimes |0_F\rangle \rightarrow (a|0_M\rangle - b|1_M\rangle)$
- $|1_F\rangle \otimes |1_F\rangle \rightarrow (a|1_M\rangle - b|0_M\rangle)$

6.3.3 Ricezione e decoding

Il signor Fabbri trasmette i due bits classici al signor Magri, il quale, in base ai bits ricevuti applica rispettivamente una tra le quattro trasformazioni I, X, Z o Y
Dopo la trasformazione del suo qubit esso si trova esattamente nello stato $|\phi\rangle = a|0\rangle + b|1\rangle$ inizialmente posseduto dal fotone del signor Fabbri, quindi è stato teletrasportato il qubit del signor Fabbri al signor Magri attraverso la trasmissione di due bits classici.

La notizia veramente strabiliante è che usando due bits classici, che possono codificare in totale solo quattro diversi possibili valori, si possono trasmettere i due coefficienti complessi a e b che possono assumere una infinità di valori.
Anche il *teletrasporto* è un elemento chiave nella sicurezza delle trasmissioni che può essere realizzata usando le tecnologie quantistiche.

6.3.4 Prime conclusioni

Con questo capitolo si è conclusa la presentazione dei concetti base dell'informatica quantistica.
Come si è visto i gates quantistici possono riprodurre completamente qualsiasi circuito logico, quindi possono in linea di principio rimpiazzare l'attuale tecnologia elettronica usata per

il calcolo automatico (computing).
Usando i principi dell' entanglement e della sovrapposizione degli stati è possibile andare oltre alla computazione classica, realizzando dei circuiti che sfruttano appieno le peculiarità della meccanica quantistica che abbiamo qui esposto.
Il risultato è di ottenere algoritmi più veloci per risolvere problemi complessi come ad esempio quello noto del commesso viaggiatore (Karthik, 2018).

L'informatica quantistica è destinata a entrare prepotentemente nella attuale società dell'informazione, aver compiuto questo primo percorso cognitivo nei suoi intimi segreti è un passo importante per restare allineati al progresso scientifico e tecnologico e per proseguire il percorso imparando anche a sfruttare le sue nuove caratteristiche.

Molte delle novità interessanti su questo argomento sono pubblicate sugli articoli scientifici come quelli riportati in bibliografia. Le nozioni qui presentate permetteranno di affrontarne la lettura.

Domanda n. 20

L'operatore $U \otimes I$ che agisce sul ket di stato di due qubits ha come effetto:

1. La trasformazione identità dell'uno e la trasformazione U dell'altro
2. La somma delle trasformazioni per entrambi i qubits
3. Trasforma il ket in un ket unitario

CAPITOLO 7

Il computer quantistico

Lo scopo di un computer quantistico è di implementare fisicamente gli stati dei qubits e le trasformazioni unitarie che operano su di essi. Con *fisicamente* si intende che deve esistere realmente, non solo in formule matematiche, uno strumento che abbia dei qubits e dei dispositivi per modificarli. Si tratta quindi di progettare e costruire un sistema *fisico* reale composto da registri di qubits e strumentazione che agisca su questi. Come ci si può aspettare questa è una sfida molto complessa sia concettuale che tecnologica. Per affrontare questa sfida trasformandola nella risoluzione di un problema con metodo analitico, il fisico DiVicenzo propose una lista di requisiti per costruire un sistema fisico capace di elaborazione quantistica, cioè un computer quantistico.

7.1 I principi di DiVincenzo

I principi sono suddivisi in due gruppi, i primi concernono la computazione:

- Un sistema di qubits ben caratterizzati e scalabile in dimensione
- La possibilità di inizializzare (impostare) lo stato dei qubits
- Tempi lungi di decoerenza (perdita di coerenza)
- Un sistema di quantum gates universale
- La possibilità di eseguire la misura di ogni qubit

I secondi riguardano la comunicazione

- Possibilità di trasformare qubits di calcolo in qubits di scambio informazioni
- Possibilità di trasmettere qubits in modo affidabile

Dall'inizio del XXI secolo ad oggi (2021) questa sfida è stata raccolta da diverse organizzazioni sia a scopo di ricerca che a scopo commerciale. Sono stati costruiti diversi prototipi funzionanti che ne hanno dimostrato la fattibilità. Ciò nonostante, la realizzazione di un computer quantistico rimane ancora una vera sfida perché ognuno dei punti della lista di DiVincenzo presenta ancora degli aspetti complessi. Vediamoli nel dettaglio.

7.2 Computer quantistico fotonico

In questo testo lo stato di un qubit è sempre stato associato allo stato di polarizzazione di un fotone. Ci sono due motivi per questa scelta. Da un lato, per quanto il fotone sia una particella priva di massa e quindi sfuggente all'intuizione comune, il concetto di polarizzazione e di lente polarizzata è molto comune, in quanto se ne fa esperienza già con gli occhiali da sole o con gli occhiali per la visione in tre dimensioni. Dall'altro, allo stato attuale della tecnologia, i computer quantistici basati sulla fotonica, sembrano quelli che possono mantenere più facilmente la promessa di essere integrati su chip e scalare in dimensione. In ogni modo, per coerenza con il resto del testo, in questo capitolo viene fornita una panoramica delle problematiche e delle possibilità della realizzazione di un computer quantistico di tipo fotonico.

Il qubit è un sistema fisico, naturale o artificiale, caratterizzato da due stati fisici ben distinti. Lo stato di polarizzazione di un fotone, ampiamente utilizzato in questo testo, è un esempio di sistema fisico a due stati, per questo motivo è possibile pensare ad un computer quantistico basato su qubit di tipo fotonico.
Allo stato attuale esistono già processori quantistici come lo Xanadu^{TM}che utilizzano questa tecnologia realizzata direttamente su chip di silicio. Per fare un primo passo nell'architet-

tura di un processore quantistico, vediamo come i criteri visti sopra sono realizzati nella sua declinazione fotonica.

- Usando i fotoni come qubits, un processore quantistico fotonico dispone già di un sistema fisico a due stati ben caratterizzati. I sistemi attuali come lo Xanadu però non usano stati di singoli fotoni, ma stati quantistici detti *squeezed*.

- Inizializzare un qubit codificato nella polarizzazione di un fotone è una sfida ben più ardua di inizializzare un bit classico. Il problema infatti sta nella sua natura intrinsecamente quantistica. L'unico modo per essere sicuri di aver prodotto un fotone è quello di rilevarlo con una apparecchio, ma in questo caso il fotone viene assorbito e quindi non è più utile come qubit. Quindi la rivelazione di un fotone con un filtro polaroid ci permette anche di conoscerne lo stato di polarizzazione ma al contempo si perde il fotone stesso. D'altra parte, la mancata rivelazione di un fotone da parte di un polaroid ci lascia in una situazione di incertezza, infatti sono possibili due alternative, l'una è che il fotone sia passato perché si trova in un dato stato di polarizzazione, l'altra è che il fotone non sia stato emesso per nulla.
  Come si può capire il problema della inizializzazione dei qubits fotonici è alquanto sfidante.
  Una possibile soluzione per preparare (inizializzare) i qubit in uno stato definito (per esempio 0 o 1) è di usare

la tecnica detta *Spontaneous parametric down-conversion* che consiste nel trasformare un fotone ad alta energia in una coppia di fotoni a più bassa energia.
La caratteristica di questo processo è che i due fotoni emessi sono entangled, quindi dalla polarizzazione di uno è possibile risalire alla polarizzazione dell'altro vedi paragrafo 3.12. In questo modo uno dei due fotoni può essere misurato e sacrificato per conoscere la polarizzazione dell'altro. A questo punto in base alla polarizzazione voluta si mantiene il fotone come è oppure si trasforma otticamente in modo da cambiarne lo stato di polarizzazione.

- Il concetto di decoerenza, o meglio di perdita di coerenza è molto complesso e difficilmente è possibile crearsene una idea intuitiva senza andare nella profondità dei calcoli. Ciò nonostante, è bene indagarlo almeno ad un livello descrittivo.
  Nel capitolo 3 è stato introdotta la funzione d'onda e si è visto che essa deve descrivere completamente il sistema considerato. Si consideri ora un sistema complesso, e suddivisibile in più sotto sistemi, per esempio possiamo pensare ad una molecola che è composta da diversi atomi. Ogni atomo è un sotto sistema del sistema molecola. Dalla meccanica quantistica possiamo aspettarci che debba esistere una funzione d'onda $\psi_{molecola}(x)$ che descriva completamente la molecola. Supponiamo ora di voler studiare un atomo della molecola. Potremmo essere indotti a pensare di studiare la funzione $\psi_{atomo}(x)$, ma questo in

generale non sarebbe corretto perché con ogni probabilità l'atomo oggetto di studio (o i suoi costituenti) si trova in entanglement con gli altri atomi, quindi non è possibile isolarne una specifica funzione d'onda per descriverlo.
Se ora si considera la reazione chimica che ha condotto gli atomi a legarsi, si troverò un istante in cui essi non erano ancora entangled e quindi potevano essere descritti singolarmente ognuno dalla propria funzione d'onda.
Un ragionamento analogo può valere per un sistema costituito da un fotone ed una serie di atomi. In pratica per un sistema complesso, i singoli costituenti non sono descrivibili in termini di funzione d'onda, esattamente come per due fotoni entangled non esistono separatamente la funzione d'onda dell'uno e quella dell'altro. Ciò detto un computer quantistico deve operare sui qubits che devono essere descritti da funzioni d'onda, quindi quando viene preparato un qubit deve essere *libero* da entanglement e in uno stato ben definito. Il problema è che in breve tempo il qubit entrerà in *contatto* con il resto dell'ambiente, ad esempio l'elettronica del computer quantistico stesso e inizierà a *interagire* e a perdere la purezza del suo stato per entrate in entanglement con il resto del sistema.
Prima che ciò avvenga il computer quantistico deve aver sfruttato il qubit. Il tempo di coerenza richiesto è quindi un compromesso tra il tempo *operativo* dei gates che devono agire sul qubit e il tempo per il quale si può supporre che il qubit non abbia interagito con l'ambiente.

Alcuni computer a *ioni intrappolati* sfruttano come qubit i due livelli energetici definiti dalla separazione *iperfine* dovuta all'interazione degli spin degli elettroni con quello nucleare. Il tempo di decoerenza di questi stati è di migliaia di anni e quindi, come si può immaginare è una ottima soluzione. Altre tecnologie sfruttano sempre ioni intrappolati ma usano uno stato fondamentale e uno stato eccitato dello ione come stati base $|0\rangle$ e $|1\rangle$. Il tempo di decoerenza in questo caso è molto breve (dell'ordine di grandezza del secondo) però sempre maggiore dei tempi *operativi* dei quantum gates.

Per quanto riguarda la realizzazione di computer fotonici bisogna tener conto che viaggiando i fotoni alla velocità della luce, il tempo operativo dei gates deve essere per forza di cose ridottissimo, il problema si riduce quindi a incanalare il fotone nel gate prima che abbia perso la sua coerenza. Questo però non è un problema di poco conto. I computer fotonici su chip attualmente in produzione non usano singoli fotoni per codificare i qubit ma stati di fotoni detti *squeezed* che permettono di gestire meglio la decoerenza.

- Un sistema di gates universale che agisca sui fotoni è in linea di principio abbastanza semplice. Le trasformazioni del qubit singolo sono realizzabili in termini di componenti di ottica lineare come specchi, beam splitter e phase shifter. Il CNOT, che insieme alle trasformazioni su qubit singolo forma un sistema di gates universale, può esse-

re realizzato sempre con le stesse componenti ma usando altri due fotoni detti *ancillari*. Sebbene l'idea di realizzare computer solo con ottica lineare sia interessante, anche perché in linea di principio si potrebbe sperimentare a casa propria con un investimento limitato, questa linea di ricerca non sembra destinata a sfociare in prodotti commerciali perché è difficilmente scalabile.
Le soluzioni più commerciali tendono a sfruttare le proprietà dell'ottica non lineare per produrre circuiti quantistici integrati su silicio.

La possibilità di trasformare un qubit di calcolo in qubit di informazione è il corrispondente quantistico del lavoro che fanno le nostre schede di rete sui bit classici quando ci colleghiamo ad internet con il PC o con il telefono. Lo stesso deve saper fare un computer quantistico se lo si vuole collegare ad un rete di computer quantistici.
In linea di principio la trasformazione tra qubit di calcolo e qubit di comunicazione arriva *gratis* nella computazione fotonica. Infatti un fotone che è stato trasformato da un gate può essere inviato così com'è lungo una linea ottica (per esempio una fibra ottica) e raggiungere un altro computer quantistico.
In teoria sembra semplice ma nella realtà le cose sono più complesse. Infatti come accennato le soluzioni fotoniche attualmente usate sono basate sugli stati squeezed per i quali il tempo di coerenza non è idealmente infinito come per un singolo fotone.

I qubit fotonici possono essere trasmessi facilmente sia *nel vuoto* che lungo una linea ottica. Il problema è ovviamente quello di mantenere lo stato di coerenza del qubit fintanto che non raggiunge la propria destinazione.

7.3 Programmazione

Quanto visto finora ha chiarito come sia possibile costruire un circuito quantistico che crea computazione, però non si è visto come fare a programmarlo, cioè stabilire la sequenza di trasformazioni quantistiche che devono aver luogo attraverso la scrittura di un programma.
Programmare un circuito quantistico significa configurare i quantum gates, per esempio stabilire l'angolo di rotazione di un operatore. Quindi a livello concreto significa stabilire una precisa sequenza di operazioni unitarie che possono avvenire su uno o più qubits.
Nel prossimo capitolo si vedrà un esempio concreto di programma quantistico, e quindi di come provare anche sperimentalmente a creare un circuito quantistico. La cosa che si vuole chiarire in questo paragrafo è cosa succede realmente quando si esegue un programma quantistico.
Una cosa che va chiarita è che ad oggi, non esiste un computer quantistico che sia completamente indipendente dalla computazione classica. Quando si esegue un programma quantistico è presente un'elettronica (computer) classica che legge ed in-

terpreta il programma. Questo è scritto usando bit classici e l'elettronica, insieme al software classico, traduce il programma in azioni di configurazione del circuito quantistico.
In pratica la programmazione in sé è classica, la configurazione del circuito è classica mentre l'esecuzione è quantistica.

CAPITOLO 8

Programmazione di un circuito quantistico

Nel capitolo 5 è stata sviluppata l'idea di *gate quantistici* e si è visto come questi possono essere usati per costruire circuiti reversibili analoghi a quelli classici, come ad esempio il circuito sommatore. Nel capitolo 6 si è utilizzata la meccanica quantistica per ottenere: non clonazione, teletrasporto e codifica densa, che non hanno un analogo classico.

Per apprezzare la differenza tra la computazione classica (o convenzionale) e quella quantistica, vediamo una semplice quanto efficace applicazione del principio di dense-coding.
Il problema che stiamo per affrontare consiste nello stabilire se una certa funzione $f(x)$, di cui non si conosce l'implementazione, sia una funzione costante oppure no.
Per ridurre al minimo la complessità della presentazione fac-

ciamo l'ipotesi semplificativa che la funzione accetti un solo bit di informazione. In tal caso possiamo distinguere i due casi, cioè costante o non costante come segue:

Costante: $f(0) = f(1) = 1$
Non costante: $f(0) = 0$, $f(1) = 1$

Se la funzione $f(x)$ è ignota, l'unica strategia con cui si può distinguere tra i due casi è quella di chiamare due volete la stessa funzione passando la prima volta come argomento 0 e la seconda volta 1. Per esempio in linguaggio C avremmo:

```
int f(int x)
{
 return 0;
}
```

per il caso costante e: int f(int x) return x;

per il caso non costante. Per testare i due casi si ha:

```
int r[2];
for(int i=0;i<=1;i++)
{
 r[i]=f(i);
}
if(r[0]==r[1])
{
//costante
}
else
```

```
{
//non costante
}
```

È chiaro che anche soluzioni del tipo:

```
int r = f(0)+f(1);
```

non cambiano il fatto che il codice della fuzione f venga eseguito per due volte.
Usando il dense coding, possiamo ridurre il numero di chiamate necessarie ad uno, quindi è possibile risolvere il problema posto dimezzando quella che viene anche chiamata *complessità di chiamata o di comunicazione (query)*. Il problema che stiamo per impostare è noto come *Deutsch problem*.

8.1 Progettazione del circuito

L'obiettivo è quello di mostrare che la computazione quantistica può risovere il Deutsch problem usando una solo chiamata a funzione. Per dimostrare questa affermazione si deve essere in grado di scrivere un programma (progettare un circuito) che implementi il problema. Al fine di mantenere la reversibiltà della computazione, se una funzione $f(x)$ è rappresentabile nella computazione classica come:

$$x \to f(x)$$

questa può essere rappresentata

$$x, y \to (x, y \oplus f(x))$$

in quella quantistica.
In pratica si usa più memoria, però dal risultato della computazione è sempre possibile risalire al valore delle variabili di input, quindi la computazione diventa reversibile. Ovviamente affiché la computazione sia reversibile è necessario conoscere la definizione della funzione f, cosa che ci accingiamo a fare.

Il passaggio appena illustrato vale tanto per un circuito quantistico quanto per uno classico, in tutti i casi se si configura la chiamata a funzione come visto sopra, questa risulterà reversibile. Vediamo ora nello specifico come eseguire detta chiamata nel mondo quantistico.
Come abbiamo visto nel capitolo 6, le operazioni che trasformano i qubit sono operatori unitari, perciò è necessario riscrivere l'azione della funzione f nei termini di un operatore unitario che chiameremo U_f.

Per il caso costante, l'operatore U_f può essere scritto come:

$$U_f = I \otimes X$$

mentre per il caso non costante si ha:

$$U_f = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes X$$

Come si può constatare facilmente, l'azione di questi due operatori su qubits che si trovino in stati $|0\rangle$ o $|1\rangle$ corrisponde esattamente all'azione della funzione f definita come:

$$x, y \rightarrow (x, y \oplus f(x))$$

Per valutare se f è costante possiamo calcolare prima $U_f(|0\rangle|0\rangle)$ dove lo stato $|0\rangle|0\rangle$ rappresenta i valori di x e y entrambi a 0, poi

$U_f(|1\rangle|0\rangle)$ dove mantenedo costante y poniamo $x = 1$. Confronatando i valori delle due espressioni si deduce se la funzione f sia o meno costante.
Se operassimo con U_f direttamente sugli stati $|0\rangle|0\rangle$ e $|1\rangle|0\rangle$ non avremmo nessun vantaggio rispetto all'implementazione classica di questo questo problema, in quanto dovremmo comunque seguire due *chiamate* o run del circuito, una per lo stato $|0\rangle|0\rangle$ e una per lo stato $|1\rangle|0\rangle$.
Se invece dallo stato iniziale $x, y = |0\rangle|0\rangle$ costruiamo lo stato $|+\rangle|-\rangle$, dove i due stati $|+\rangle$ e $|-\rangle$ definiti come:

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

e

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

vediamo che è sufficiente eseguire una singola trasformazione U_f, cioè l'analogo di una chiamata ad f per stabilire se la funzione f sia costante oppure no. Infatti si ha che:

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

e quindi dopo l'azione di U_f si agisce con l'operatore di Hadamard H su x e poi si misura il valore di x che potrà essere o 0 oppure 1. Nel primo caso la funzione è costante, nel secondo caso la funzione non è costante.

8.2 Codice QASM

Nel capitolo 5 abbiamo visto come rappresentare graficamente le trasformazioni quantistiche, vediamo ora che esse possono anche essere codificate in un linguaggio detto *assembly quantistico* o appunto QASM.
Il QASM è il tentativo di realizzare una sorta di standard per i circuiti quantistici, ma i tempi sono ancora giovani, quindi è meglio aspettare e vedere cosa succederà. Intanto, per gli anni in cui viene scritta questa edizione del testo, il codice qui riportato può essere eseguito su un computer quantistico della IBM o anche sul simulatrore Qiskit, entrambi ambienti accessibili gratuitamente.

```
OPENQASM 2.0;
include "qelib1.inc";
gate nG0 ( param ) q1, q2  {
//***f(q1) ccostante***
//id q1;
//x q2;
//***f(q1) NON-ccostante***
cx q1,q2;
}

qreg q[2];
creg c[1];

id q[0];
```

```
x q[1];
h q[0];
h q[1];
nG0(0) q[0],q[1];
h q[0];
measure q[0] -> c[0];
```

Analizziamo il codice. La dichiarazione gate nG0 (param) q1, q2 è analoga alla dichiarazione di una fuzione il cui nome sia nG0, che operi sull'argomento definito dai due qubit q0 e q1 e che sia parametrizzata da param. In questo contesto non analizziamo il significato di questo parametro che riguarda un argomento più avanzato che non viene discusso in questo testo. In realtà gate nG0 (param) q1, q2 non è una funzione ma un operatore, e rappresenta esattamente l'operatore U_f che abbiamo definito nei conti eseguiti poco sopra.
L'operatore (gate) gate nG0 è quindi la nostra *black-box* di cui dobbiamo stabilire se l'implementazione sia di tipo costante o non costante.
La dichiarazione qreg q[2] inizializza un registro quantistico a due qubit nello stato $|0\rangle$, mentre creg c[1] indica un registro classico ad un solo bit usato per registrare il risultato di una misura quantistica.

8.3 Esecuzione del programma

L'esecuzione del programma termina con la misura del qubit q[0] il cui risultato viene memorizzato sul registro classico e

risulta quindi essere un consueto bit di un sistema informatico convenzionale (un PC per esempio). Valutando il valore di c si può conoscere il risultato della computazione quantistica.
Se c risulta essere uguale a 0 la funzione f è costante, mentre, se risulta uguale ad 1, la funzione non è costante.
Per provare il codice e testare le due eventualità, si commenti l'una o l'altra sezione della definizione del gate nG0.

8.4 Conclusioni

Concludiamo questo testo con una domanda:

La computazione quantistica prenderà mai il posto di quella classica?

Probabilmente questa domanda sta spaventando molti, sia tra i programmatori che tra i grandi della produzione industriale. È ovvio che l'idea che una nuova tecnologia possa prendere il posto di quella a cui si è dedicato energie e risorse non sia piacevole.
Penso però che chi si occupa di informatica classica possa stare relativamente tranquillo.
Come scrisse Landau, la formulazione stessa della Meccanica Quantistica è intrinsecamente impossibile senza l'inclusione della meccanica classica, e penso che lo stesso valga per la computazione quantistica.
L'informatica quantistica sfrutta caratteristiche della fisica che non sono apprezzabili nel mondo classico in cui vive l'essere umano, per questo credo sarà necessario anche in futuro di-

sporre di sistemi informatici classici per *trasportare* i risultati quantistici nel mondo classico.

Tirando le somme, è probabile che la computazione quantistica si svilupperà molto in futuro, ma anche che l'informatica classica rimarrà al suo fianco, allo stesso modo in cui la fisica classica continua a servire da supporto a quella quantistica.

8.4.1 Come approfondire gli argomenti

Gli argomenti che sono stati trattati in questo testo sono tutti piuttosto complessi e meritevoli ognuno di uno studio dedicato. Chi fosse interessato ad approfondirli per completare la propria conoscenza può iniziare leggendo i libri e gli articoli riportati nel capitolo finale della bibliografia. In base alla propria preparazione di partenza potrà trovare i riferimenti citati come utili o difficoltosi, in ogni caso costituiscono un punto di partenza. Buon proseguimento.

8.5 Risposte alle domande

In alcuni e-reader, la numerazione delle risposte possibili alle domande compare come 1, 2, 3 anziché a), b), c). In quel caso, si interpretino le risposte: a come 1, b come 2 e c come 3.

- **1** - a.
- **2** - b.
- **3** - b.
- **4** - a.

- **5** - b.
- **6** - a.
- **7** - a.
- **8** - b.
- **9** - c.
- **10** - c.
- **11** - c.
- **12** - c.
- **13** - b.
- **14** - a.
- **15** - a.
- **16** - c.
- **17** - a.
- **18** - a.
- **19** - b.
- **20** - a.

CAPITOLO 9

Appendice: Esercizi in linguaggio C

Negli esercizi che seguono è richiesto di implementare una emulazione dei circuiti quantistici visti nel testo.

Gli esercizi devono essere svolti usando i tipi di dato e le funzioni definite nella libreria libSSQ.
La libreria permette di emulare la creazione di stati di un singolo qubit, due e tre qubits.
Allo stesso modo, si creano gli operatori tensoriali che agiscono sugli stati di qubits.

Per i primi tre esercizi è proposta una soluzione, mentre la soluzione del quarto e del quinto è lasciata al lettore che potrà

postarla sul gruppo facebook **Reti Neurali in C** all'indirizzo https://www.facebook.com/groups/1229389267230026/

Attenzione: per quanto si possa conoscere a fondo la programmazione in linguaggio C, questi esercizi possono risultare molto *sfidanti*, si consiglia di prendersi il tempo dovuto per analizzare bene la libreria proposta per capirne i principi ispiratori e il modello sottostante.

Il codice riportato è presente anche su GitHub all'indirizzo https://github.com/francescosisini/ nel repository LIBRO-Informatica-Quantistica

9.1 Libreria libSSQ

L'organizzazione in tipi e funzioni proposta nel codice che segua ha lo scopo di ricalcare il più fedelmente possibile i concetti matematici visti nel calcolo tensoriale.
Il codice non è realizzato per massimizzare né l'efficienza né la semplicità d'uso, ma proprio per aderire quanto più fedelmente possibile alla realtà matematica e fisica fin qui introdotta.

9.1.1 Kets

Per rappresentare uno stato fisico di qubits si usano i tipi ket1 per i singoli qubit e ket2 e ket3 per gli stati di due e tre qubits rispettivamente.
Per creare un ket di un singolo qubit si usa la funzione crea_ket1

a cui si passano per argomento i coefficienti delle ampiezze degli stati della base base1 che è un tipo struct che ha due campi e1 ed e2 di tipo elemento1.

In prima istanza è possibile che si faccia fatica a recepire il modello C presentato nella libreria, ma con un po' di pazienza, ci si accorgerà che esso è più intuitivo di quanto appaia all'inizio. Per rompere il ghiaccio vediamo un primo esempio in cui si crea lo stato di un qubit di valore zero espresso rispetto alla base standard.

```
double complex a = 1;
double complex b = 1;
double complex norm = csqrt(1/(a*a+b*b));
ket1 psi = crea_ket1(a*nomr,b*norm,base_std1);
```

Si vede dalla prima riga del codice che la libreria usa le macro definite in complex.h per le operazioni sui numeri complessi.
La variabile base_std1 è predefinita nella libreria e rappresenta un'istanza del tipo base1. La chiamata alla funzione crea_ket1 ritorna une ket1 di ampiezze a e b riferite agli elementi e1 ed e2 della base standard che è una variabile predefinita nella libreria e può essere usata nel codice utente dichiarandola come extern

9.1.2 Kets per due e tre qubits

Se si prova a stampare a video i membri x e y del ket ψ si vede che essi corrispondono esattamente ai due parametri a e b passati per argomento alla funzione:

```
printf("%g+i%g,%g+i%g->",psi.x, psi.y);
```

Quindi esiste una relazione uno a uno tra i parametri passati e i coefficienti dei ket di base usati per definire lo stato. Analogamente, per creare un ket di stato di due qubits si devono passare quattro parametri a, b, c e d che corrispondono alle due ampiezze di ognuno dei due ket:

```
double complex a = 1;
double complex b = 0;
double complex c = 1;
double complex d = 0;

double complex norm = csqrt(1/(a*a+b*b+c*c+d*d));

a = a*norm;
b = b*norm;
c = c*norm;
d = d*norm;

ket2 psi2 = crea_ket2(a,b,c,d,base_std2);

printf("(%g+i%g,%g+i%g)x(%g+i%g,%g+i%g) = ",a, b,c,d);
```

```
printf("%g+i%g,%g+i%g,%g+i%g,%g+i%g\n\n",
psi2.x11, psi2.x12,psi2.x21, psi2.x22);
```

Si noti che, diversamente dal caso di un solo qubit, i coefficienti x11, x12, x21 e x22 dei quattro ket di base usati per definire lo stato dei due qubits sono diversi dei coefficienti dei due qubit considerati singolarmente (vedi paragrafo 5.4).

Per creare il ket di stato di tre qubits si usa la funzione

```
ket3
crea_ket3(double complex x1, double complex y1,
        double complex x2, double complex y2,
        double complex x3, double complex y3,
        base3 b);
```

il cui prototipo è analogo a quelli visti per creare stati di due o un qubit. Si rammenti che in questo caso i parametri che definiscono gli stati dei singoli qubits sono sei mentre i coefficienti di base sono otto (2^3).

Operatori

Gli operatori sono definiti in modo simile ai ket di stato.
Per quelli che devono operare sui ket di tipo ket1 viene definito un tipo di dato per rappresentare gli elementi dello spazio tensoriale $V \otimes V^*$. Il nuovo tipo è chiamato elemento1_1 , dove i due indici *1* e *_1* indicano che l'operatore ha un elemento dello spazio V e uno dello spazio V^*:

```
typedef struct
```

```
  {
    double complex c[2][2];
  } elemento1_1;
```

Essi sono rappresentati dai tipi o*n_n* dove l'indice *n* può essere sostituito con il numero 1,2 o 3, quindi gli operatori che agiscono su stati di un singolo qubit sono rappresentati dal tipo o*1_1* e similmente i tipi o*2_2* e o*3_3* rappresentano gli operatori che agiscono su stati di due e tre qubits rispettivamente. L'indice *n* si riferisce ai ket mentre *_n* si riferisce ai bra.
Ogni operatore *o1_1* è definito rispetto alla base base1_1 costituita da quattro elementi di tipo elemento1_1 come segue:

```
   typedef struct
{
  elemento1_1 e1e_1; //00
  elemento1_1 e1e_2; //01
  elemento1_1 e2e_1; //10
  elemento1_1 e2e_2; //11


} base1_1;
```

La libreria libSSQ provvede una istanza già predefinita del tipo base1_1 attraverso la variabile base_std1_1

```
const base1_1 base_std1_1 =
  {
    {
      {1,0,
      0,0}
```

```
    }
    ,
    {
      {0,1,
      0,0}
    }
    ,
    {
      {0,0,
      1,0}
    }
    ,
    {
      {0,0,
      0,1}
    }
  };
```

La variabile può essere usata dichiarandola extern nel codice.

Le base standard base_std1_1 è definita in modo che il prodotto dei suoi elementi con quelli della base standard dei ket1, base_std1, soddisfi le relazioni definite nel paragrafo 3.3 (Prodotto fra duali e vettori).

Per creare un operatore si usa la funzione

```
op1_1 crea_tensore1_1
(double complex a11,double complex
a12,double complex a21,
double complex a22, base1_1 b)
```

Per esempio, l'operatore identità viene creato come segue:

```
op1_1 o = crea_tensore1_1(1,0,0,1,base_std1_1);
```

Una volta creato un operatore si può usare per trasformare uno ket usando la funzione ket1 trasforma_ket1(op1_1 o,ket1 ket) come nell'esempio che segue:

```
ket1 psip = trasforma_ket1(o,psi);
```

Dopo la breve descrizione della libreria e del suo uso si raccomanda ora di studiare il codice nel dettaglio e poi passare senza indugio agli esercizi del paragrafo seguente.

libSSQ.h

```
/**************************************************************************
 *
 * libSSQ @2020 Scuola Sisini
 * Licenza GPL3
 *
    This program is free software: you can redistribute it and/or modify
    it under the terms of the GNU General Public License as published by
    the Free Software Foundation, either version 3 of the License, or
    (at your option) any later version.

    This program is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public License
    along with this program.  If not, see <https://www.gnu.org/licenses/>.
 */

#pragma once
#ifndef LIBSSQ_H
#define LIBSSQ_H

#include <complex.h>

/*__________________SINGOLO QUBIT_________________*/

/*-------------------------------------------------------
 *
 * Elemento e in  V
 */
typedef struct
```

```
{
  double complex c[2];
} elemento1;

/*_____________________________________________________
 *
 * Base qubit singolo
 */
typedef struct
{
  elemento1 e1;
  elemento1 e2;

} base1;



/*_____________________________________________________
 *
 * Un ket di ampiezze complesse x e y sulla base b
 */
typedef struct
{
  double complex x,y;
  base1 b;
} ket1;


/*_________DOPERATORE TENSORIALE_1X1___________*/

/*_____________________________________________________
 *
 * Elemento  ee* in V(x)V*
 */
typedef struct
{
  double complex c[2][2];
} elemento1_1;

/*_____________________________________________________
 *
 * Base tensoori V@V*
 */
typedef struct
{
  elemento1_1 e1e_1; //00
  elemento1_1 e1e_2; //01
  elemento1_1 e2e_1; //10
  elemento1_1 e2e_2; //11


} base1_1;

/*_____________________________________________________
```

```
 *
 * Un operatore sullo spazio 11
 * di componenti a11, a12, a21, a22 sull base b
 *
 */
typedef struct
{
  double complex a11,a12,a21,a22;
  base1_1 b;
} op1_1;


/*_____________________DOPPIO QUBIT_____________________*/

/*_____________________________________________________
 *
 * Elemento  ee in V(x)V
 */
typedef struct
{
  double complex c[4];
} elemento2;

/*_____________________________________________________
 *
 * Base 2 qubits
 */
typedef struct
{
  elemento2 e1f1; //00
  elemento2 e1f2; //01
  elemento2 e2f1; //10
  elemento2 e2f2; //11


} base2;

/*_____________________________________________________
 *
 * Un ket di 2 qubit di ampiezze complesse x1 e y1
 * e x2, y2  sulla base b
 *
 */
typedef struct
{
  double complex x11,x12,x21,x22;
  base2 b;
} ket2;

/*__________DOPERATORE TENSORIALE_2X2_____________*/

/*_____________________________________________________
 *
 * Elemento  efe*f* in V(x)V(x)V*(x)V*
```

```
 */
typedef struct
{
  double complex c[4][4];
} elemento2_2;

/*_____________________________________________________
 *
 * Base tensoori V@V*
 */
typedef struct
{
  //-riga 1
  elemento2_2 e1f1e_1f_1;
  elemento2_2 e1f1e_1f_2;
  elemento2_2 e1f1e_2f_1;
  elemento2_2 e1f1e_2f_2;
  //-riga 2
  elemento2_2 e1f2e_1f_1;
  elemento2_2 e1f2e_1f_2;
  elemento2_2 e1f2e_2f_1;
  elemento2_2 e1f2e_2f_2;
  //-riga 3
  elemento2_2 e2f1e_1f_1;
  elemento2_2 e2f1e_1f_2;
  elemento2_2 e2f1e_2f_1;
  elemento2_2 e2f1e_2f_2;
  //-riga 4
  elemento2_2 e2f2e_1f_1;
  elemento2_2 e2f2e_1f_2;
  elemento2_2 e2f2e_2f_1;
  elemento2_2 e2f2e_2f_2;

} base2_2;

/*_____________________________________________________
 *
 * Un operatore sullo spazio 11
 * di componenti a11, a12, a21, a22 sull base b
 *
 */
typedef struct
{
  double complex a11,a12,a13,a14,
    a21,a22,a23,a24,
    a31,a32,a33,a34,
    a41,a42,a43,a44;
  base2_2 b;
} op2_2;


/*____________________TRIPLO QUBIT_________________*/

/*_____________________________________________________
```

```
 *
 * Elemento  eee in V(x)V(x)V
 */
typedef struct
{
  double complex c[8];
} elemento3;

/*_____________________________________________________
 *
 * Base 3 qubits
 */
typedef struct
{
  elemento3 e1f1g1; //000
  elemento3 e1f1g2; //001
  elemento3 e1f2g1; //010
  elemento3 e1f2g2; //011
  elemento3 e2f1g1; //100
  elemento3 e2f1g2; //101
  elemento3 e2f2g1; //110
  elemento3 e2f2g2; //111

} base3;

/*_____________________________________________________
 *
 * Un ket di 3 qubit di ampiezze complesse x1 e y1
 * , x2, y2 e x3, y3  sulla base b
 *
 */
typedef struct
{
  double complex x111,x112,x121,x122,x211,x212,x221,x222;
  base3 b;
} ket3;

/*_____DOPERATORE TENSORIALE_3X3_____*/

/*_____________________________________________________
 *
 * Elemento  efe*f* in V(x)V(x)V*(x)V*
 */
typedef struct
{
  double complex c[8][8];
} elemento3_3;

/*_____________________________________________________
 *
 * Base tensoori V@V@V@V*@V*@V*
 */
typedef struct
{
```

```
//-riga 1
elemento3_3 e1f1g1e_1f_1g_1;
elemento3_3 e1f1g1e_1f_1g_2;
elemento3_3 e1f1g1e_1f_2g_1;
elemento3_3 e1f1g1e_1f_2g_2;
elemento3_3 e1f1g1e_2f_1g_1;
elemento3_3 e1f1g1e_2f_1g_2;
elemento3_3 e1f1g1e_2f_2g_1;
elemento3_3 e1f1g1e_2f_2g_2;
 //-riga 2
elemento3_3 e1f1g2e_1f_1g_1;
elemento3_3 e1f1g2e_1f_1g_2;
elemento3_3 e1f1g2e_1f_2g_1;
elemento3_3 e1f1g2e_1f_2g_2;
elemento3_3 e1f1g2e_2f_1g_1;
elemento3_3 e1f1g2e_2f_1g_2;
elemento3_3 e1f1g2e_2f_2g_1;
elemento3_3 e1f1g2e_2f_2g_2;
 //-riga 3
elemento3_3 e1f2g1e_1f_1g_1;
elemento3_3 e1f2g1e_1f_1g_2;
elemento3_3 e1f2g1e_1f_2g_1;
elemento3_3 e1f2g1e_1f_2g_2;
elemento3_3 e1f2g1e_2f_1g_1;
elemento3_3 e1f2g1e_2f_1g_2;
elemento3_3 e1f2g1e_2f_2g_1;
elemento3_3 e1f2g1e_2f_2g_2;
 //-riga 4
elemento3_3 e1f2g2e_1f_1g_1;
elemento3_3 e1f2g2e_1f_1g_2;
elemento3_3 e1f2g2e_1f_2g_1;
elemento3_3 e1f2g2e_1f_2g_2;
elemento3_3 e1f2g2e_2f_1g_1;
elemento3_3 e1f2g2e_2f_1g_2;
elemento3_3 e1f2g2e_2f_2g_1;
elemento3_3 e1f2g2e_2f_2g_2;

//-riga 1
elemento3_3 e2f1g1e_1f_1g_1;
elemento3_3 e2f1g1e_1f_1g_2;
elemento3_3 e2f1g1e_1f_2g_1;
elemento3_3 e2f1g1e_1f_2g_2;
elemento3_3 e2f1g1e_2f_1g_1;
elemento3_3 e2f1g1e_2f_1g_2;
elemento3_3 e2f1g1e_2f_2g_1;
elemento3_3 e2f1g1e_2f_2g_2;
 //-riga 2
elemento3_3 e2f1g2e_1f_1g_1;
elemento3_3 e2f1g2e_1f_1g_2;
elemento3_3 e2f1g2e_1f_2g_1;
elemento3_3 e2f1g2e_1f_2g_2;
elemento3_3 e2f1g2e_2f_1g_1;
elemento3_3 e2f1g2e_2f_1g_2;
elemento3_3 e2f1g2e_2f_2g_1;
```

```
  elemento3_3 e2f1g2e_2f_2g_2;
   //--riga 3
  elemento3_3 e2f2g1e_1f_1g_1;
  elemento3_3 e2f2g1e_1f_1g_2;
  elemento3_3 e2f2g1e_1f_2g_1;
  elemento3_3 e2f2g1e_1f_2g_2;
  elemento3_3 e2f2g1e_2f_1g_1;
  elemento3_3 e2f2g1e_2f_1g_2;
  elemento3_3 e2f2g1e_2f_2g_1;
  elemento3_3 e2f2g1e_2f_2g_2;
   //--riga 4
  elemento3_3 e2f2g2e_1f_1g_1;
  elemento3_3 e2f2g2e_1f_1g_2;
  elemento3_3 e2f2g2e_1f_2g_1;
  elemento3_3 e2f2g2e_1f_2g_2;
  elemento3_3 e2f2g2e_2f_1g_1;
  elemento3_3 e2f2g2e_2f_1g_2;
  elemento3_3 e2f2g2e_2f_2g_1;
  elemento3_3 e2f2g2e_2f_2g_2;

} base3_3;

/*-------------------------------------------------------
 *
 * Un operatore sullo spazio 111
 * di componenti a11, a12,...,a88 sull base b
 *
 */
typedef struct
{
  double complex
  a11,a12,a13,a14,a15,a16,a17,a18,
    a21,a22,a23,a24,a25,a26,a27,a28,
    a31,a32,a33,a34,a35,a36,a37,a38,
    a41,a42,a43,a44,a45,a46,a47,a48,
    a51,a52,a53,a54,a55,a56,a57,a58,
    a61,a62,a63,a64,a65,a66,a67,a68,
    a71,a72,a73,a74,a75,a76,a77,a78,
    a81,a82,a83,a84,a85,a86,a87,a88;
  base3_3 b;
} op3_3;



/*-------------Funzioni---------------------------------*/

/*-------------------------------------------------------
 *
 * crea un ket di stato di ampiezze complesse
 * x e y sulla base b
 */
ket1
crea_ket1(double complex x, double complex y, base1 b);
```

```
/*_____________________________________________________
 *
 * crea un ket di stato di due qubits ampiezze complesse
 * x1, y1, x2 e y2 sulla base b
 * come prodotto dei due key
 * non     possibile creare direttamente ket entangled
 */
ket2
crea_ket2(double complex x1, double complex y1,
          double complex x2, double complex y2,base2 b);

/*_____________________________________________________
 *
 * crea un ket di stato di tre qubits ampiezze complesse
 */
ket3
crea_ket3(double complex x1, double complex y1,
          double complex x2, double complex y2,
          double complex x3, double complex y3,
          base3 b);




/*_____________________________________________________
 *
 * crea un operatore (tensore 1_1) di coefficienti
 * complessi a11, a12, a21 e a22 sulla base b
 */
op1_1 crea_tensore1_1
(double complex a11,double complex a12,double complex a21,double complex a22, base1_1 b);

/*_____________________________________________________
 *
 * crea un operatore (tensore 2_2) di coefficienti
 * complessi a11, a12,...,a44 sulla base b
 */
op2_2 crea_tensore2_2
(double complex a11,double complex a12,double complex a13,double complex a14,
 double complex a21,double complex a22,double complex a23,double complex a24,
 double complex a31,double complex a32,double complex a33,double complex a34,
 double complex a41,double complex a42,double complex a43,double complex a44,
 base2_2 b
 );

/*_____________________________________________________
 *
 * crea un operatore (tensore 3_3) di coefficienti
 * complessi a11, a12,...,a44 sulla base b
 */
op3_3 crea_tensore3_3
(double complex a11,double complex a12,double complex a13,double complex a14,
 double complex a15,double complex a16,double complex a17,double complex a18,
```

```
double complex a21,double complex a22,double complex a23,double complex a24,
double complex a25,double complex a26,double complex a27,double complex a28,

double complex a31,double complex a32,double complex a33,double complex a34,
double complex a35,double complex a36,double complex a37,double complex a38,

double complex a41,double complex a42,double complex a43,double complex a44,
double complex a45,double complex a46,double complex a47,double complex a48,

double complex a51,double complex a52,double complex a53,double complex a54,
double complex a55,double complex a56,double complex a57,double complex a58,

double complex a61,double complex a62,double complex a63,double complex a64,
double complex a65,double complex a66,double complex a67,double complex a68,

double complex a71,double complex a72,double complex a73,double complex a74,
double complex a75,double complex a76,double complex a77,double complex a78,

double complex a81,double complex a82,double complex a83,double complex a84,
double complex a85,double complex a86,double complex a87,double complex a88,

base3_3 b
);



/*-------------------------------------------------------
 *
 * Trasforma un ket tramite un operatore
 *
 */
ket1 trasforma_ket1(op1_1 o,ket1 ket);

ket2 trasforma_ket2(op2_2 o,ket2 ket);

ket3 trasforma_ket3(op3_3 o,ket3 ket);

#endif
```

libSSQ.c

```
/*****************************************************************************
 *
 * libSSQ @2020 Scuola Sisini
 * Licenza GPL3
 *
    This program is free software: you can redistribute it and/or modify
    it under the terms of the GNU General Public License as published by
    the Free Software Foundation, either version 3 of the License, or
    (at your option) any later version.

    This program is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
```

```
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public License
    along with this program.  If not, see <https://www.gnu.org/licenses/>.
 */

#include <complex.h>
#include <stdio.h>
#include "libSSQ.h"

/*-------------------------------------------------------
 *
 * Base STANDARD qubit singolo
 */
const base1 base_std1 =
  {
    {{1,0}},//0
    {{0,1}} //1
  };

/*-------------------------------------------------------
 *
 * Base STANDARD 2qubits
 */
const base2 base_std2=
  {
    {{1,0,0,0}},//00
    {{0,1,0,0}},//01
    {{0,0,1,0}},//10
    {{0,0,0,1}} //11
  };

/*-------------------------------------------------------
 *
 * Base STANDARD 3qubits
 */
const base3 base_std3=
  {
    {{1,0,0,0,0,0,0,0}},//00
    {{0,1,0,0,0,0,0,0}},//00
    {{0,0,1,0,0,0,0,0}},//00
    {{0,0,0,1,0,0,0,0}},//00

    {{0,0,0,0,1,0,0,0}},//00
    {{0,0,0,0,0,1,0,0}},//00
    {{0,0,0,0,0,0,1,0}},//00
    {{0,0,0,0,0,0,0,1}},//00

  };



/*-------------------------------------------------------
 *
```

```
 * Base STANDARD opertori 1_1
 */
const base1_1 base_std1_1 =
  {
    {
      {1,0,
      0,0}
    }
    ,
    {
      {0,1,
      0,0}
    }
    ,
    {
      {0,0,
      1,0}
    }
    ,
    {
      {0,0,
      0,1}
    }
  };

/*-------------------------------------------------
 *
 * Base STANDARD opertori 2_2
 */
const base2_2 base_std2_2 =
  {
    {
      {1,0,0,0,
       0,0,0,0,
       0,0,0,0,
       0,0,0,0}
    }
    ,
    {
      {0,1,0,0,
       0,0,0,0,
       0,0,0,0,
       0,0,0,0}
    }
    ,
    {
      {0,0,1,0,
       0,0,0,0,
       0,0,0,0,
       0,0,0,0}
    }
    ,
    {
      {0,0,0,1,
```

```
    0,0,0,0,
    0,0,0,0,
    0,0,0,0}
}
,
{
  {0,0,0,0,
   1,0,0,0,
   0,0,0,0,
   0,0,0,0}
}
,
{
  {0,0,0,0,
   0,1,0,0,
   0,0,0,0,
   0,0,0,0}
}
,
{
  {0,0,0,0,
   0,0,1,0,
   0,0,0,0,
   0,0,0,0}
}
,
{
  {0,0,0,0,
   0,0,0,1,
   0,0,0,0,
   0,0,0,0}
}
,
{
  {0,0,0,0,
   0,0,0,0,
   1,0,0,0,
   0,0,0,0}
}
,
{
  {0,0,0,0,
   0,0,0,0,
   0,1,0,0,
   0,0,0,0}
}
,
{
  {0,0,0,0,
   0,0,0,0,
   0,0,1,0,
   0,0,0,0}
}
,
```

```
    {
      {0,0,0,0,
       0,0,0,0,
       0,0,0,1,
       0,0,0,0}
    }
    ,
    {
      {0,0,0,0,
       0,0,0,0,
       0,0,0,0,
       1,0,0,0}
    }
    ,
    {
      {0,0,0,0,
       0,0,0,0,
       0,0,0,0,
       0,1,0,0}
    }
    ,
    {
      {0,0,0,0,
       0,0,0,0,
       0,0,0,0,
       0,0,1,0}
    }
    ,
    {
      {0,0,0,0,
       0,0,0,0,
       0,0,0,0,
       0,0,0,1}
    }
  };

/*-------------------------------------------------------
 *
 * Base STANDARD opertori 3_3
 */
const base3_3 base_std3_3 =
  {
    1,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,

    0,1,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
```

```
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,1,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,1,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,1,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,1,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,1,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,1,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
```

```
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
1,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,1,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,1,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,1,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,1,0,0,0,
0,0,0,0,0,0,0,0,

```

```
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,1,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,1,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
1,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,1,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,1,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,1,0,0,0,0,
```

```
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,1,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,1,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,1,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,1,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
1,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
```

```
0,1,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,1,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,1,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,1,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,1,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
```

```
0,0,0,0,0,0,0,1,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
1,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,1,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,1,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,1,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
```

```
0,0,0,0,0,0,0,0,
0,0,0,0,0,1,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,1,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,1,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
1,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,1,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,1,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
```

```
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,1,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,1,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,1,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,1,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,1,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
1,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
```

```
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,1,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,1,0,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,1,0,0,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,1,0,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,1,0,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
```

```
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,1,
0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
1,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,1,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,1,0,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,1,0,0,0,0,

0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,
```

```
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,1,0,0,

    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,1,0,0,

    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,1
  };


//---Servizio
void matrice_x_vettore2
(double complex v_out[2],elemento1_1 m,elemento1 v_in)
{
  for(int i=0;i<2;i++) v_out[i] = 0;

  for(int i=0;i<2;i++)
    {
      for(int j=0;j<2;j++)
        {
          v_out[i]+=m.c[i][j]*v_in.c[j];
        }
    }
}
void matrice_x_vettore4
(double complex v_out[4],elemento2_2 m,elemento2 v_in)
{
  for(int i=0;i<4;i++) v_out[i] = 0;

  for(int i=0;i<4;i++)
    {
      for(int j=0;j<4;j++)
        {
          v_out[i]+=m.c[i][j]*v_in.c[j];
        }
    }
}
```

```
void matrice_x_vettore8
(double complex v_out[8],elemento3_3 m,elemento3 v_in)
{
  for(int i=0;i<8;i++) v_out[i] = 0;

  for(int i=0;i<8;i++)
    {
      for(int j=0;j<8;j++)
        {
          v_out[i]+=m.c[i][j]*v_in.c[j];
        }
    }
}


/*-------------------------------------------------------
 *
 * crea un ket di stato di ampiezze complesse
 * x e y sulla base b
 */
ket1
crea_ket1(double complex x, double complex y, base1 b)
{
  ket1 ket = {x,y,b};
  return ket;
}

/*-------------------------------------------------------
 *
 * crea un ket di stato di due qubits ampiezze complesse
 * x1, y1, x2 e y2 sulla base b
 * come prodotto dei due key
 * non     possibile creare direttamente ket entangled
 */
ket2
crea_ket2(double complex x1, double complex y1,
          double complex x2, double complex y2,base2 b)
{
  ket2 ket ={x1*x2,x1*y2,y1*x2,y1*y2,b};
  return ket;

}

/*-------------------------------------------------------
 *
 * crea un ket di stato di tre qubits ampiezze complesse
 */
ket3
crea_ket3(double complex x1, double complex y1,
          double complex x2, double complex y2,
          double complex x3, double complex y3,
          base3 b)
{
  ket3 ket = {
```

```
    x1*x2*x3,
    x1*x2*y3,
    x1*y2*x3,
    x1*y2*y3,

    y1*x2*x3,
    y1*x2*y3,
    y1*y2*x3,
    y1*y2*y3,b
  };
    return ket;

}

/*_____________________________________________________
 *
 * crea un operatore (tensore 1_1) di coefficienti
 * complessi a11, a12, a21 e a22 sulla base b
 */
op1_1 crea_tensore1_1
(double complex a11,double complex a12,double complex a21,double complex a22, base1_1 b)
{
  op1_1 o = {a11,a12,a21,a22,b};
  return o;
}

/*_____________________________________________________
 *
 * crea un operatore (tensore 2_2) di coefficienti
 * complessi a11, a12,...,a44 sulla base b
 */
op2_2 crea_tensore2_2
(double complex a11,double complex a12,double complex a13,double complex a14,
 double complex a21,double complex a22,double complex a23,double complex a24,
 double complex a31,double complex a32,double complex a33,double complex a34,
 double complex a41,double complex a42,double complex a43,double complex a44,
 base2_2 b
 )
{
  op2_2 o={a11,a12,a13,a14,a21,a22,a23,a24,a31,a32,a33,a34,a41,a42,a43,a44,b};
  return o;
}

/*_____________________________________________________
 *
 * crea un operatore (tensore 3_3) di coefficienti
 * complessi a11, a12,...,a88 sulla base b
 */
op3_3 crea_tensore3_3
(double complex a11,double complex a12,double complex a13,double complex a14,
 double complex a15,double complex a16,double complex a17,double complex a18,

 double complex a21,double complex a22,double complex a23,double complex a24,
 double complex a25,double complex a26,double complex a27,double complex a28,
```

```
 double complex a31,double complex a32,double complex a33,double complex a34,
 double complex a35,double complex a36,double complex a37,double complex a38,

 double complex a41,double complex a42,double complex a43,double complex a44,
 double complex a45,double complex a46,double complex a47,double complex a48,

 double complex a51,double complex a52,double complex a53,double complex a54,
 double complex a55,double complex a56,double complex a57,double complex a58,

 double complex a61,double complex a62,double complex a63,double complex a64,
 double complex a65,double complex a66,double complex a67,double complex a68,

 double complex a71,double complex a72,double complex a73,double complex a74,
 double complex a75,double complex a76,double complex a77,double complex a78,

 double complex a81,double complex a82,double complex a83,double complex a84,
 double complex a85,double complex a86,double complex a87,double complex a88,

 base3_3 b
 )
{
  op3_3 o ={
    a11, a12, a13, a14,
    a15, a16, a17, a18,

    a21, a22, a23, a24,
    a25, a26, a27, a28,

    a31, a32, a33, a34,
    a35, a36, a37, a38,

    a41, a42, a43, a44,
    a45, a46, a47, a48,

    a51, a52, a53, a54,
    a55, a56, a57, a58,

    a61, a62, a63, a64,
    a65, a66, a67, a68,

    a71, a72, a73, a74,
    a75, a76, a77, a78,

    a81, a82, a83, a84,
    a85, a86, a87, a88,
    b
  };

  return o;
}

/*_____________________________________________________
 *
```

```
 * Trasforma un ket tramite un operatore
 *
 */
ket1 trasforma_ket1(op1_1 o,ket1 ket)
{
  ket1 ketp;

  //Preparazione array di servizio
  double complex v[2];
  elemento1 e[2] = {ket.b.e1,ket.b.e2};
  elemento1_1 ee_[2][2] = {o.b.e1e_1,o.b.e1e_2,o.b.e2e_1,o.b.e2e_2};
  double complex x[2] =  {ket.x,ket.y};
  double complex a[2][2] = {o.a11,o.a12,o.a21,o.a22};

  //Inizializzazione
  ketp.x = 0;
  ketp.y = 0;
  for(int k=0;k<2;k++)
    for(int i=0;i<2;i++)
      {
        //Il tensore o    sviluppato sulla base eie_j come:
        //o = a11*e1e_1+...+a22*e2e_2
        //Il ket ket    sviluppato sulla base ei come:
        //ket = x*e1+y*e2
        //Il prodottotto o*ket si sviluppa linearmente come:
        //o*ket=(a11*e1e_1+...+a22*e2e_2)*(x*e1+y*e2) quindi:
        //o*ket=e1e_1(e1)*a11*x+... dove si ricorda che
        //e1e_1(e1)    un prodotto matrice x colonna
        for(int j=0;j<2;j++)
          {
            matrice_x_vettore2(v,ee_[i][j],e[k]);
            ketp.x += v[0]*a[i][j]*x[k];
            ketp.y += v[1]*a[i][j]*x[k];
          }
      }
  ketp.b = ket.b;
  return ketp;
}

ket2 trasforma_ket2(op2_2 o,ket2 ket)
{
  ket2 ketp;
  ketp.x11 = ketp.x12 = ketp.x21 = ketp.x22 = 0;
  //Preparazione array di servizio
  double complex v[4];
  elemento2 e[4] = {ket.b.e1f1,ket.b.e1f2,ket.b.e2f1,ket.b.e2f2};
  elemento2_2 ee_[4][4] = {
    o.b.e1f1e_1f_1,o.b.e1f1e_1f_2,o.b.e1f1e_2f_1,o.b.e1f1e_2f_2,
    o.b.e1f2e_1f_1,o.b.e1f2e_1f_2,o.b.e1f2e_2f_1,o.b.e1f2e_2f_2,
    o.b.e2f1e_1f_1,o.b.e2f1e_1f_2,o.b.e2f1e_2f_1,o.b.e2f1e_2f_2,
    o.b.e2f2e_1f_1,o.b.e2f2e_1f_2,o.b.e2f2e_2f_1,o.b.e2f2e_2f_2
  };
  double complex x[4] =  {ket.x11,ket.x12,ket.x21,ket.x22};
  double complex a[4][4] = {
```

```
    o.a11,o.a12,o.a13,o.a14,
    o.a21,o.a22,o.a23,o.a24,
    o.a31,o.a32,o.a33,o.a34,
    o.a41,o.a42,o.a43,o.a44
  };

   for(int k=0;k<4;k++)
    for(int i=0;i<4;i++)
      {
        //Per una spiegazione, vedi i commenti di trasforma_ket1
        for(int j=0;j<4;j++)
          {
            matrice_x_vettore4(v,ee_[i][j],e[k]);

            ketp.x11 += v[0]*a[i][j]*x[k];
            ketp.x12 += v[1]*a[i][j]*x[k];
            ketp.x21 += v[2]*a[i][j]*x[k];
            ketp.x22 += v[3]*a[i][j]*x[k];
          }
      }
   ketp.b = ket.b;
   return ketp;
}

ket3 trasforma_ket3(op3_3 o,ket3 ket)
{
  ket3 ketp;
  ketp.x111 = ketp.x112 = ketp.x121 = ketp.x122 =
    ketp.x211 = ketp.x212 = ketp.x221 = ketp.x222 = 0;

  //Preparazione array di servizio
  double complex v[8];

  elemento3 e[8] = {
    ket.b.e1f1g1,
    ket.b.e1f1g2,
    ket.b.e1f2g1,
    ket.b.e1f2g2,
    ket.b.e2f1g1,
    ket.b.e2f1g2,
    ket.b.e2f2g1,
    ket.b.e2f2g2};

  elemento3_3 ee_[8][8] = {
    o.b.e1f1g1e_1f_1g_1,
    o.b.e1f1g1e_1f_1g_2,
    o.b.e1f1g1e_1f_2g_1,
    o.b.e1f1g1e_1f_2g_2,
    o.b.e1f1g1e_2f_1g_1,
    o.b.e1f1g1e_2f_1g_2,
    o.b.e1f1g1e_2f_2g_1,
    o.b.e1f1g1e_2f_2g_2,

    o.b.e1f1g2e_1f_1g_1,
```

```
o.b.e1f1g2e_1f_1g_2,
o.b.e1f1g2e_1f_2g_1,
o.b.e1f1g2e_1f_2g_2,
o.b.e1f1g2e_2f_1g_1,
o.b.e1f1g2e_2f_1g_2,
o.b.e1f1g2e_2f_2g_1,
o.b.e1f1g2e_2f_2g_2,

o.b.e1f2g1e_1f_1g_1,
o.b.e1f2g1e_1f_1g_2,
o.b.e1f2g1e_1f_2g_1,
o.b.e1f2g1e_1f_2g_2,
o.b.e1f2g1e_2f_1g_1,
o.b.e1f2g1e_2f_1g_2,
o.b.e1f2g1e_2f_2g_1,
o.b.e1f2g1e_2f_2g_2,

o.b.e1f2g2e_1f_1g_1,
o.b.e1f2g2e_1f_1g_2,
o.b.e1f2g2e_1f_2g_1,
o.b.e1f2g2e_1f_2g_2,
o.b.e1f2g2e_2f_1g_1,
o.b.e1f2g2e_2f_1g_2,
o.b.e1f2g2e_2f_2g_1,
o.b.e1f2g2e_2f_2g_2,

//----------------

o.b.e2f1g1e_1f_1g_1,
o.b.e2f1g1e_1f_1g_2,
o.b.e2f1g1e_1f_2g_1,
o.b.e2f1g1e_1f_2g_2,
o.b.e2f1g1e_2f_1g_1,
o.b.e2f1g1e_2f_1g_2,
o.b.e2f1g1e_2f_2g_1,
o.b.e2f1g1e_2f_2g_2,

o.b.e2f1g2e_1f_1g_1,
o.b.e2f1g2e_1f_1g_2,
o.b.e2f1g2e_1f_2g_1,
o.b.e2f1g2e_1f_2g_2,
o.b.e2f1g2e_2f_1g_1,
o.b.e2f1g2e_2f_1g_2,
o.b.e2f1g2e_2f_2g_1,
o.b.e2f1g2e_2f_2g_2,

o.b.e2f2g1e_1f_1g_1,
o.b.e2f2g1e_1f_1g_2,
o.b.e2f2g1e_1f_2g_1,
o.b.e2f2g1e_1f_2g_2,
o.b.e2f2g1e_2f_1g_1,
o.b.e2f2g1e_2f_1g_2,
o.b.e2f2g1e_2f_2g_1,
o.b.e2f2g1e_2f_2g_2,
```

```
    o.b.e2f2g2e_1f_1g_1,
    o.b.e2f2g2e_1f_1g_2,
    o.b.e2f2g2e_1f_2g_1,
    o.b.e2f2g2e_1f_2g_2,
    o.b.e2f2g2e_2f_1g_1,
    o.b.e2f2g2e_2f_1g_2,
    o.b.e2f2g2e_2f_2g_1,
    o.b.e2f2g2e_2f_2g_2,
      };

  double complex x[8] =  {
    ket.x111,
    ket.x112,
    ket.x121,
    ket.x122,
    ket.x211,
    ket.x212,
    ket.x221,
    ket.x222
  };

  double complex a[8][8] = {
    o.a11,o.a12,o.a13,o.a14,o.a15,o.a16,o.a17,o.a18,
    o.a21,o.a22,o.a23,o.a24,o.a25,o.a26,o.a27,o.a28,
    o.a31,o.a32,o.a33,o.a34,o.a35,o.a36,o.a37,o.a38,
    o.a41,o.a42,o.a43,o.a44,o.a45,o.a46,o.a47,o.a48,

    o.a51,o.a52,o.a53,o.a54,o.a55,o.a56,o.a57,o.a58,
    o.a61,o.a62,o.a63,o.a64,o.a65,o.a66,o.a67,o.a68,
    o.a71,o.a72,o.a73,o.a74,o.a75,o.a76,o.a77,o.a78,
    o.a81,o.a82,o.a83,o.a84,o.a85,o.a86,o.a87,o.a88
  };

   for(int k=0;k<8;k++)
    for(int i=0;i<8;i++)
      {
        //Per una spiegazione, vedi i commenti di trasforma_ket1
        for(int j=0;j<8;j++)
          {
            matrice_x_vettore8(v,ee_[i][j],e[k]);

            ketp.x111 += v[0]*a[i][j]*x[k];
            ketp.x112 += v[1]*a[i][j]*x[k];
            ketp.x121 += v[2]*a[i][j]*x[k];
            ketp.x122 += v[3]*a[i][j]*x[k];
            ketp.x211 += v[4]*a[i][j]*x[k];
            ketp.x212 += v[5]*a[i][j]*x[k];
            ketp.x221 += v[6]*a[i][j]*x[k];
            ketp.x222 += v[7]*a[i][j]*x[k];
          }

      }
   ketp.b = ket.b;
```

```
    return ketp;
}
```

9.1.3 Esercizi

Si risolvano i seguenti esercizi usando la libreria definita nel paragrafo precedente. Il codice con le soluzioni degli esercizi è presentato nel paragrafo successivo.

1. Si consideri un qubit definito da un fotone che si trovi nello stato $|0\rangle$. Si agisca con l'operatore σ_x di Pauli. Qual'è ora lo stato in cui si trova il qubit?
2. Si implementi il gate CNOT e se ne verifichi l'azione sullo stato: $|1\rangle \otimes |1\rangle$ e sullo stato $|0\rangle \otimes |1\rangle$.
3. Si faccia riferimento al paragrafo 6.2 e si implementi la trasmissione di un solo qubit per comunicare una informazione che ha quattro possibili valori, come l'esempio dei quattro punti cardinali presente nel testo.
4. Si realizzi un circuito full-adder reversibile.
5. Si implementi una comunicazione basata sul teletrasporto quantistico.

Note di compilazione

Le soluzioni proposte sono scritte in C standard e compilano sotto ogni architettura e sistema operativo. Per la compilazione sotto Linux si ricorda di linkare la libreria matematica:

```
gcc -o e3 libSSQ.c esercizio_3.c -lm
```

Usando il compilatore GCC l'uso dei caratteri di formattazione %g genera un innocuo messaggio di warning.

9.2 Soluzioni

1. **Soluzione**: Il testo dell'esercizio fa riferimento ad un singolo qubit quindi per svolgere l'esercizio dovremo usare il tipo ket1.
   L'operatore σ_x può essere costruito ricordando la sua forma matriciale definita nel paragrafo 5.2. Gli elementi della matrice devono essere passati come argomento alla funzione {crea_tensore1_1} come segue:

```
#include
#include
#include
#include "libSSQ.h"

extern base1 base_std1;
extern base1_1 base_std1_1;


int main()
{
  printf("____QUBIT␣SINGOLO___\n");

  double complex a = 1;
  double complex b = 0;
  double complex norm =csqrt( 1/(a*a+b*b));

  ket1 psi = crea_ket1(a*norm,b*norm,base_std1);

  op1_1 o = crea_tensore1_1(0,1,1,0,base_std1_1);

  ket1 psip = trasforma_ket1(o,psi);

  printf("\nOperatore␣sigma_x:\n",psi.x, psi.y);
  printf("%g+i%g,%g+i%g␣->␣",psi.x, psi.y);
  printf("%g+i%g,%g+i%g\n",psip.x, psip.y);

}
```

   Compilando ed eseguendo il codice si vede che il ket di stato passa da $|0\rangle$ a $|1\rangle$, quindi il qubit passa da 0 ad 1.

2. **Soluzione**: Il gate CNOT ha due qubit di ingresso e due

di uscita, quindi per rappresentare lo stato di ingresso è necessario usare il tipo ket2.

L'operatore CNOT può essere costruito usando la sua forma matriciale definita nel paragrafo 5.4. Gli elementi della matrice devono essere passati come argomento alla funzione crea_tensore2_2 come segue:

```
#include
#include
#include
#include "libSSQ.h"

extern base2 base_std2;
extern base2_2 base_std2_2;


int main()
{
  printf("_____2 QUBITs____\n");

  double complex a = 1;
  double complex b = 0;
  double complex c = 1;
  double complex d = 0;

  double complex norm1 =
    csqrt(1/(a*conj(a)+b*conj(b)));
  double complex norm2 =
    csqrt(1/(c*conj(c)+d*conj(d)));

  ket2 psi2 = crea_ket2(a*norm1,b*norm1,c*norm2,d*norm2,base_std2);

  op2_2 o2 = crea_tensore2_2(
                        1,0,0,0,
                        0,1,0,0,
                        0,0,0,1,
                        0,0,1,0,
                        base_std2_2);

 ket2  psip2 = trasforma_ket2(o2,psi2);

  printf("Operatore CNOT:\n");
  printf("%g+i%g,%g+i%g,%g+i%g,%g+i%g -> ",psi2.x11, psi2.x12,psi2.x21, psi2.x22);
  printf("%g+i%g,%g+i%g,%g+i%g,%g+i%g\n",psip2.x11, psip2.x12,psip2.x21, psip2.x22);

  a = 0;
  b = 1;
  c = 0;
  d = 1;
```

```
  norm1 =
    csqrt(1/(a*conj(a)+b*conj(b)));
  norm2 =
    csqrt(1/(c*conj(c)+d*conj(d)));

  psi2 = crea_ket2(a*norm1,b*norm1,c*norm2,d*norm2,base_std2);

  psip2 = trasforma_ket2(o2,psi2);


  printf("Operatore␣CNOT:\n");
  printf("%g+i%g,%g+i%g,%g+i%g,%g+i%g␣->␣",psi2.x11, psi2.x12,psi2.x21, psi2.x22);
  printf("%g+i%g,%g+i%g,%g+i%g,%g+i%g\n",psip2.x11, psip2.x12,psip2.x21, psip2.x22);

}
```

Compilando ed eseguendo il codice si vede che il ket di stato passa da $|0\rangle$ a $|1\rangle$, quindi il qubit passa da 0 ad 1.

3. **Soluzione**: il primo passaggio per risolvere questo esercizio è di creare uno stato entangled. È stato spiegato nella teoria che uno stato entangled è tale perché non può essere scritto come il prodotto di due stati singoli, per tale motivo non è possibile usare la funzione crea_ket2, ma il ket deve essere creato specificando manualmente le sue componenti.

   Usando l'operatore σ_x si deve trasformare il qubit del signor Fabbri in modo da codificare il numero 1, cioè il SUD (con riferimento all'esempio della teoria).

   Per decodificare lo stato si deve operare in successione con il gate CNOT e il gate $H \otimes I$, le cui componenti devono essere calcolate sviluppando il prodotto tensoriale dell'operare H e I.

```
#include
#include
#include
#include "libSSQ.h"
```

```
extern base2 base_std2;
extern base2_2 base_std2_2;


int main()
{
  printf("_____Dense Coding____\n\n");

  printf("1)Viene creato e condiviso uno stato entangled |00>+|11> tra Fabbri e Magri");
  double complex norm = 1/sqrt(2);
  //Non pu  essere costruito come il prodotto di due ket
  ket2 ent_k = {1*norm,0,0,1*norm, base_std2};

  //Operatore XxI
  op2_2 o2 = crea_tensore2_2(
                          0,1,0,0,
                          1,0,0,0,
                          0,0,0,1,
                          0,0,1,0,
                          base_std2_2);

 ket2  psip2 = trasforma_ket2(o2,ent_k);

 printf("\n\n2)Fabbri codifica il valore 1 (SUD) usando l'operatore XxI:\n\n");

 printf("\t%g+i%g,%g+i%g,%g+i%g,%g+i%g -> ",ent_k.x11, ent_k.x12, ent_k.x21, ent_k.x22);
 printf("\t%g+i%g,%g+i%g,%g+i%g,%g+i%g\n",psip2.x11, psip2.x12, psip2.x21, psip2.x22);

 printf("\n3)Fabbri invia il proprio qubit a Magri\n");

 printf("\n4)Magri applica un operatore CNOT e uno HxI:\n\n");
 //Passo1: trasforma con Cnot
 o2 = crea_tensore2_2(
                          1,0,0,0,
                          0,1,0,0,
                          0,0,0,1,
                          0,0,1,0,
                          base_std2_2);

 ket2 psipp2 = trasforma_ket2(o2,psip2);

 printf("\t%g+i%g,%g+i%g,%g+i%g,%g+i%g -> ",psip2.x11, psip2.x12, psip2.x21, psip2.x22);
 printf("\t%g+i%g,%g+i%g,%g+i%g,%g+i%g\n",psipp2.x11, psipp2.x12, psipp2.x21, psipp2.x22);

 double complex h = norm*1;
 o2 = crea_tensore2_2(
                          h,0,h,0,
                          0,h,0,h,
                          h,0,-h,0,
                          0,h,0,-h,
                          base_std2_2);

 ket2 psippp2 = trasforma_ket2(o2,psipp2);
```

```
printf("\t%g+i%g,%g+i%g,%g+i%g,%g+i%g␣->␣",psipp2.x11, psipp2.x12, psipp2.x21, psipp2.x22);
printf("\t%g+i%g,%g+i%g,%g+i%g,%g+i%g\n",psippp2.x11, psippp2.x12, psippp2.x21, psippp2.x22);
printf("\n5)Magri␣ha␣ricevuto␣|01>␣cio ␣SUD\n\n");


}
```

Compilando ed eseguendo il codice si vede che il ket entangled passa dallo stato $|00\rangle + |11\rangle$ allo stato $|01\rangle$ come atteso.

Bibliografia

[1] PAM Dirca, The Lagrangian in Quantum Mechanics. Physikalische Zeitschift der Sowjetunion (1933)

[2] Laundau, L., Lifsits, E. Meccanica quantistica, teoria non relativistica. Editori riuniti university press.

[3] Sisini, F. Sisini, V Informatica quantistica: introduzione con esempi in linguaggio C. Scuola Sisini (2020).

[4] A. Einstein, B. Podolsky, and N. Rosen. Can Quantum-Mechanical Description of Physical Reality Be Considered Complete? Phys. Rev. 47 (1935)

[5] Über einen die Erzeugung und Verwandlung des Lichtes betreffenden heuristischen Gesichttspunkt ("On a Heuristic Viewpoint Concerning the Production and Transformation of Light"). Annalen der Physik, 4 (1905)

[6] Heisenberg W.K. (1958), Physics and philosophy, New York Harper and Row. (Original published in 1955: Das Naturbild der heutigen Physik). Italiano: H.W.K. Natura e fisica moderna, Garzanti, 1960, pagg. 24 e 25.

[7] Shannon E. A Mathematical Theory of Communication, Bell System Technical Journal (1948)

[8] Turing, A.M. On Computable Numbers, with an Application to the Entscheidungsproblem, Proceedings of the London Mathematical Society (1937).

SCUOLA SISINI

ALTRI LIBRI DI SCUOLA SISINI

GLI AUTORI PERCORRONO PASSO A PASSO TUTTO L'ESPERIMENTO DI FUKUSHIMA TRASCINANDO IL LETTORE IN UN MONDO AFFASCINANTE DOVE BIOLOGIA, MATEMATICA ED INFORMATICA SI FONDONO IL TESTO ANALIZZA OGNI PASSAGGIO NEL DETTAGLIO, .

HTTPS://WWW.AMAZON.IT/RETI-NEURALI-NON-SUPERVISIONATE-COGNITRONE/DP/1798929244/REF

SCUOLA SISINI

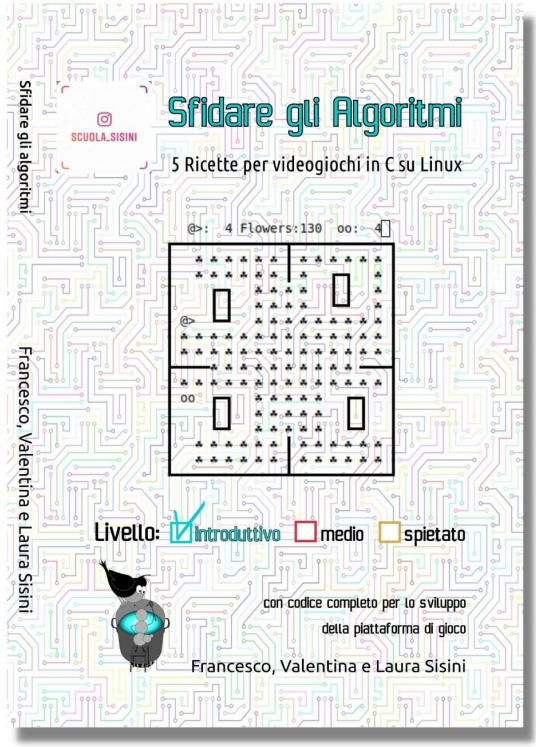
ALTRI LIBRI DI SCUOLA SISINI

HTTPS://WWW.AMAZON.IT/GP/PRODUCT/ B08GTJ2KHM/REF

IL LIBRO PROPONE UNA INTRODUZIONE GENTILE ALL'INTELLIGENZA ARTIFICIALE. NEL DETTAGLIO IMPARERAI COSA È UN AGENT, COSA È L'AMBIENTE, LE BASI DELLA TEORIA DEI GRAFI, IN CHE MODO UN AGENT RAPPRESENTA LE INFORMAZIONI CHE HA DELL'AMBIENTE IN UN GRAFO, IN CHE MODO UN AGENT COSTRUISCE UN GRAFO DELL'AMBIENTE TRAMITE L'APPRENDIMENTO PER ESPERIENZA, E CHE UNA RICERCA SU UN GRAFO È UNA TECNICA DI RAGIONAMENTO AUTOMATICO

SCUOLA SISINI

ALTRI LIBRI DI SCUOLA SISINI



HTTPS://WWW.AMAZON.IT/GP/PRODUCT/1695109325REF

IL LIBRO INTRODUCE L'USO DI ALGORITMI E DEI GRAFI PER LA SOLUZIONE DEI GIOCHI , GIOCHI CHE TRATTANO ARGOMENTI ATTUALI DELLA GAMIFICATION E DEL MONDO LAVORATIVO, LE PROBLEMATICHE AFFRONTATE SONO QUELLE DELL'INTELLIGENZA ARTIFICIALE.
TOTALMENTE BASATO SU GRAFICA A TERMINALE PER ESSERE CHIARO ANCHE AI PRINCIPIANTI . COMPLETO DI CODICI E SPIEGAZIONI

SCUOLA SISINI

ALTRI LIBRI DI SCUOLA SISINI

LA COMPUTAZIONE QUANTISTICA È BASATA SUI QUBITS CHE RAPPRESENTANO L'ANALOGO QUANTISTICO DEI BITS. I QUBITS PERMETTONO DI RISOLVERE IN MODO SEMPLICE PROBLEMI ANNOSI DELLA COMPUTAZIONE CLASSICA. QUESTO TESTO RAPPRESENTA IL PUNTO DI INGRESSO PER CHI VUOLE CAPIRE COSA SONO I QUBITS E LA LORO RELAZIONE CON I BIT CLASSICI.

HTTPS://WWW.AMAZON.IT/QUBITS-PRINCIPI-FONDAMENTALI-FRANCESCO-SISINI/DP/B08WZH5429/REF