# UNIVERSITÀ DEGLI STUDI DI NAPOLI
# "FEDERICO II"

**Scuola Politecnica e delle Scienze di Base**

**Area Didattica di Scienze Matematiche Fisiche e Naturali**

**Dipartimento di Fisica "Ettore Pancini"**


*Laurea Triennale in Fisica*

# Algoritmi di ricerca quantistica


**Relatori:**
Procolo Lucignano

**Candidato:**
Vincenzo Volpe
Matr. $N85001488$


**Anno Accademico 2018/2019**

# Indice

# Capitolo 1

# Prefazione

Feynman nel 1981 sostenne l'impossibilità di simulare efficientemente sistemi quantistici su un computer classico, data l'enorme mole di operazioni necessarie a calcolarne le proprietà spettrali e dinamiche. Egli affermò anche che a tale scopo servisse un computer basato sulle leggi della *meccanica quantistica* (QM).

L'idea di quantum computer moderna nacque nel 1985 dal fisico David Deutsch. Egli cercò, nella fisica teorica, una soluzione alla "Tesi di Church-Turing", ossia:"Ogni processo algoritmico può essere simulato efficientemente usando una macchina probabilistica di Turing". Per fare ciò teorizzò una macchina capace di simulare un sistema fisico; quindi essendo la nostra realtà basata sulle leggi della QM anche tale dispositivo lo era.

Nel 1997 vi fu un primo traguardo importante per questa nascente branca di ricerca. Il fisico Peter Shor pubblicò un articolo nel quale descriveva due algoritmi con complessità computazionale polinomiale per la risoluzione di 2 grandi problemi: la fattorizzazione in numeri primi di un intero e il problema del "logaritmo discreto". Questa scoperta puntò i riflettori sulla *quantum computation* (QC) perchè portò, oltre all'innovazione teorica, anche un differente approccio a branche scientifiche già ben consolidate, come ad esempio la crittografia.

Vi furono molti altri algoritmi sviluppati in quegli anni che ribadirono il potenziale della QC. Nel 1995 il fisico Lov Grover dimostrò che il problema della ricerca di un elemento in uno spazio disordinato ha una risoluzione più rapida su un computer quantistico. Grover, nel suo algoritmo, sviluppò una tecnica molto importate per la QC ossia l' "*amplitude amplification*".

A seguito di questi contributi pioneristici l'informazione e la computazione quantistica hanno avuto un enorme impatto e, ad oggi, esistono diversi dispositivi sperimentali in grado di risolvere algoritmi quantistici sebbene, in genere, solo per sistemi di piccola taglia. Esistono diversi paradigmi di calcolo quantistico: la gate (o universal) quantum computation (GQC), l'adiabatic quantum computation (AQC) e la topological quantum computation (TQC). In questo lavoro studiere-

mo due algoritmi quantistici per il problema della ricerca basati sui primi due approcci.

Uno di questi è l'algoritmo di Grover che è basato sull'applicazione successiva di diverse porte logiche quantistiche ad uno stato di input, in perfetta analogia formale con qualsiasi algoritmo classico. Descriveremo in dettaglio questa procedura nel paragrafo 2.1.

Il secondo approccio al problema della ricerca è basato sulla computazione quantistica adiabatica. L'idea di base è di codificare la soluzione del problema desiderata nello stato fondamentale di un'hamiltoniana interagente. Si costruisce quindi una dinamica "lenta" che all'istante iniziale parte da una configurazione di ground state del nostro sistema quantistico che sia di semplice studio, come ad esempio lo stato fondamentale di un sistema non interagente avente lo stesso numero di gradi di libertà del problema che si vuole risolvere. Il teorema adiabatico ci assicura che mediante un evoluzione controllata il sistema rimarrà costantemente nello stato fondamentale e non verrà promosso a livelli energetici superiori. Quindi, all'istante finale, lo stato del mio sistema quantistico codifica la soluzione del problema studiato. I dettagli sono nel paragrafo 2.2.

In alcuni casi, come quello del problema di searching, sono stati creati algoritmi sia in GQC sia in AQC. Si dimostra che se per un problema esistono algoritmi sia per la GQC che per la AQC allora questi avranno la stessa complessità computazionale.

I *qbit*, in analogia ai bit classici, sono gli elementi su cui si basa la QC. Queste componenti possono essere create in vari modi [1][cap.7] e di seguito sono riportati brevemenți alcuni di essi:

1. Fotoni ottici. Questi possono essere trasportati per lunghe distanze attraverso fibre ottiche senza perdere informazioni, si possono [modificare semplicemente] attraverso un cambio di fase e si possono combinare semplicemente mediando un *beamsplitters*.

2. Trappole ioniche. Tale metodo sfrutta gli spin di un piccolo gruppo di atomi carichi. Questi ultimi sono isolati e intrappolati mediante trappole elettromagnetiche successivamente, per modificare il loro stato di spin, si utilizza una luce monocromatica.

3. Risonanza di nuclei magnetici (NMR). Tale metodo sfrutta la manipolazione degli stati di spin nucleari attraverso la frequenza di onde radio.

4. Circuiti superconduttivi. Si usano piccoli anelli percorsi da supercorrenti. A seconda del verso della corrente c'è un flusso concatenato all'anello che punta nelle due direzioni opposte lungo la normale al piano contenente l'anello. I due stati di flusso, che in un anello superconduttivo sono quantizzati, rappresentano i due valori della variabile dicotomica.

Possiamo considerare i qbit come una generalizzazione dei bit. Questi potranno si codificare un informazione binaria ma solo se "perdono" la loro natura probabilistica. Infatti ad un qbit sono associate delle probabilità di essere o in uno stato o nell'altro. In prima approssimazione possimo considerare un qbit come una moneta, essa ha due facce ma quando la lanciamo questa inizia a ruotare e sinchè non si ferma avremo $1/2$ di possibilita che sia testa o croce. Formalizzeremo meglio i qbit nel capitolo 1.1.

Un aspetto su cui bisogna soffermarsi è il tempo di *decoerenza*. I costituenti dei qbit, come visto, sono tutti elementi fisici, atomi, molecole, fotoni ecc... Per preservare l'informazione che contengono si cerca di isolare tali componenti dal mondo esterno. Questo viene fatto perchè, per esempio, se un atomo interagisce con un campo elettromagnetico esterno, non voluto dallo sperimentatore, potrebbe cambiare il proprio stato. Il tempo medio dopo il quale un qbit perde lo stato codificato è chiamato tempo di decoerenza. Per spiegare meglio tale fenomeno riportiamo in seguito un analogia con sistemi classici. Consideriamo un Hard disk composto da un disco metallico e un punta che, mediante la generazione di un campo elettromagnetico, polarizza le varie sezioni di questa piastra le quali assumeranno il ruolo di bit . Consideriamo uno di questi ultimi che inizialmete sia nello stato 0, passato del tempo questo avrà, a causa dei campi magnetici ambientali, una probailità p di cambiare il proprio stato e una probabilità 1-p di rimanere nello stesso. Allo stesso modo l'interazione fra il mondo circostante e i qbit modifica le probabilità associate ai vari stati e rende l'azione di misura imprevedibile.

Nel paragrafo successivo procediamo con la formulazione matematica di qbit, gate e circuiti che ci serviranno poi per descrivere gli algoritmi di searching.

## 1.1 Qbit

La teoria dell'informazione "classica" è basata sulla possibilità di preparare, modificare e misurare, stati di una variabile dicotomica detti *bit*. Tipicamente, quindi, un computer classico è una collezione di sistemi fisici che possono stare in soli due stati fisicamente distinguibili come ad esempio: un interruttore che può essere apero o chiuso, lo stato di saturazione o interdizione di un transistor o le due polarizzazioni di un magnete. Analogamente i *qbit* sono le unità fondamentali su cui si basa la *quantum computation*.
Si sceglie a tale scopo un sistema fisico avente due stati $|0\rangle$ e $|1\rangle$, per semplicità espositiva consideriamo i livelli energetici di un elettrone che ruota intorno ad un atomo. In questo caso $|0\rangle$ rappresenta lo stato fondamentale e $|1\rangle$ uno stato eccitato, mediante un fascio luminoso è possibile far cambiare stato all'elettrone che quindi può essere promosso $|0\rangle \rightarrow |1\rangle$, o retrocesso $|1\rangle \rightarrow |0\rangle$. Inoltre possiamo indicare questi ultimi mediante una combinazione lineare dei due stati, chiamata *Sovrapposizione*:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \tag{1.1}$$

$|0\rangle\, e\, |1\rangle$ sono conosciuti come *stati della base computazionale*. Tali vettori formano una base ortonormale dello spazio, ossia $\langle i|j\rangle = \delta_{ij}$. Un'altra possibile rappresentazione di tale base, chiamata base canonica, che utilizzeremo in seguito è la seguente:

$$|0\rangle = \begin{bmatrix}1\\0\end{bmatrix}; |1\rangle = \begin{bmatrix}0\\1\end{bmatrix} \tag{1.2}$$

è evidente quindi l'analogia formale con gli spinori a 2 componenti che in fisica si usano per rappresentare gli stati $\uparrow$ e $\downarrow$ di uno spin 1/2. Infatti tutte le possibili realizzazioni sperimentali di qbit sono basate su sistemi quantistici la cui dinamica possa essere descritta analogamente a quella di una collezione di spin (o pseudospin) 1/2.

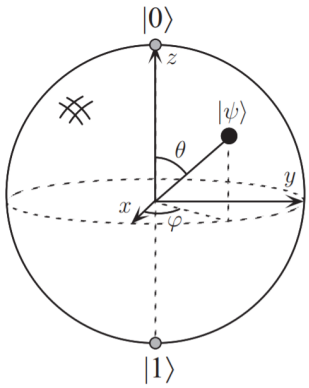
$\alpha$ e $\beta$ sono due coefficienti complessi il cui modulo quadro da la probabilità di misurare rispettivamente lo stato $|0\rangle$ o lo stato $|1\rangle$. Data questa loro natura devono rispettare la condizione di normalizzazione:

$$|\alpha|^2 + |\beta|^2 = 1 \tag{1.3}$$

Diamo di seguito una dimostrazione della 1.3 partendo dalla condizione di normalizzazione della funzione $|\psi\rangle$:

$$\begin{aligned} 1 &= \langle\psi|\psi\rangle \\ &= (\alpha^* \langle 0| + \beta^* \langle 1|)(\alpha\,|0\rangle + \beta\,|1\rangle) \\ &= |\alpha|^2 \langle 0|0\rangle + |\beta|^2 \langle 1|1\rangle + \alpha^*\beta \langle 0|1\rangle + \beta^*\alpha \langle 1|0\rangle \\ &= |\alpha|^2 + |\beta|^2 \end{aligned} \tag{1.4}$$

Figura 1.1: Sfera di Bloch




essendo la base ortonormale.
Due valori che verificano la 1.3 sono $\alpha = e^{i\gamma}\cos\frac{\theta}{2}$ e $\beta = e^{i\gamma}e^{i\varphi}\sin\frac{\theta}{2}$, sostituendoli nella 1.1 abbiamo:

$$|\psi\rangle = e^{i\gamma}(\cos\frac{\theta}{2}|0\rangle + e^{i\varphi}\sin\frac{\theta}{2}|1\rangle) \tag{1.5}$$

dove $e^{i\gamma}$ è una fase che non ha significato fisico e che possiamo trascurare:

$$|\psi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\varphi}\sin\frac{\theta}{2}|1\rangle \tag{1.6}$$

L'equazione 1.6 definisce lo stato di un qbit in temini di due parametri. La *sfera di Bloch*(Figura 1.1) rappresenta geometricamente le infinite sovrapposizioni di tale parametrizzazione.

Potremmo pensare erroneamente che, essendo infinite le sovrapposizioni, un *qbit* possa portare in sè un infinità di possibili informazioni. Tale tesi è errata, infatti quando viene effettuata una misurazione di un osservabile esso collassa nell'autostato misurato. Ciò significa che prima di valutare lo stato del nostro qbit avremo una probabilità, pari a $|\alpha|^2$, di trovare lo stato $|0\rangle$ e una probabilità, pari a $|\beta|^2$, di trovare lo stato $|1\rangle$, dopo la misura il qbit non sarà più in una *sovrapposizione* e collasserà o in $|0\rangle$ o in $|1\rangle$.

### 1.1.1 Qbit multipi

Se si hanno a disposizione n *qbit* gli elementi che compongono la base saranno $2^n$, ossia il numero di possibili combinazioni di 2 elementi in n posti. Riducendoci al caso di due qbit gli stati possibili sono $|00\rangle$,$|01\rangle$,$|10\rangle$,$|11\rangle$, tale coppia di qbit si potrà quindi trovare in una sovrapposizione dei quattro stati prima indicati:

$$|\psi\rangle = \alpha_{00}\,|00\rangle + \alpha_{01}\,|01\rangle + \alpha_{10}\,|10\rangle + \alpha_{11}\,|11\rangle \tag{1.7}$$

I coefficienti $\alpha_{ij}$ sono le ampiezze e, come nel caso unidimensionale, devono essere normalizzate $|\alpha_{00}|^2 + |\alpha_{01}|^2 + |\alpha_{01}|^2 + |\alpha_{11}|^2 = 1$ (dimostrazione analoga alla 1.4).
Per n qbit non esiste una visualizzazione geometrica analoga alla sfera di Bloch. Per qbit multipli possiamo effettuare una misura parziale su un sottoinsieme di questi. Per esempio dati 2 *qbit* supponiamo di misurare il primo e che esso collassi in $|0\rangle$, la sovrapposizione $|\psi\rangle$ diventerà:

$$|\psi\rangle = \frac{\alpha_{00}\,|00\rangle + \alpha_{01}\,|01\rangle}{\sqrt{|\alpha_{00}|^2 + |\alpha_{01}|^2}} \tag{1.8}$$

dove $\frac{1}{\sqrt{|\alpha_{00}|^2+|\alpha_{01}|^2}}$ è stato aggiunto per conservare la condizione di normalizzazione:

$$\begin{aligned}
\langle\psi|\psi\rangle &= \Big(\frac{\alpha_{00}\,\langle 00| + \alpha_{01}\,\langle 01|}{\sqrt{|\alpha_{00}|^2 + |\alpha_{01}|^2}}\Big)\Big(\frac{\alpha_{00}\,|00\rangle + \alpha_{01}\,|01\rangle}{\sqrt{|\alpha_{00}|^2 + |\alpha_{01}|^2}}\Big) \\
&= \frac{1}{|\alpha_{00}|^2 + |\alpha_{01}|^2}(|\alpha_{00}|^2\,\langle 0|0\rangle + |\alpha_{01}|^2\,\langle 1|1\rangle + \alpha_{00}\alpha_{01}\,\langle 0|1\rangle + \alpha_{01}\alpha_{00}\,\langle 1|0\rangle) \\
&= \frac{|\alpha_{00}|^2 + |\alpha_{01}|^2}{|\alpha_{00}|^2 + |\alpha_{01}|^2} = 1
\end{aligned} \tag{1.9}$$

come volevasi dimostrare. L'equazione 1.8 indica che una volta fatta la misura lo stato collassa in un sottospazio bidimensionale.

Generalizzando a n qbit, la base canonica associata ai nostri ket $\{(1,0,0,...,0),(0,1,0,...,0),...,(0,0,0,...,1)\}$ si ottiene per prodotto esterno degli stati a singolo qbit. Portiamo come esempio il caso di tre *qbit* ed, in particolare, dello stato $|101\rangle$:

$$|101\rangle = \begin{bmatrix}0\\1\end{bmatrix} \otimes \begin{bmatrix}1\\0\end{bmatrix} \otimes \begin{bmatrix}0\\1\end{bmatrix} = \begin{bmatrix}0*1*0\\0*1*1\\0*0*0\\0*0*1\\1*1*0\\1*1*1\\1*0*0\\1*0*1\end{bmatrix} = \begin{bmatrix}0\\0\\0\\0\\0\\1\\0\\0\end{bmatrix} \tag{1.10}$$

Un'ulteriore notazione per indicare gli stati di qbit multipli è quella di scriverli sostituendo alla stringa di numeri binari il corrispondente valore decimale, per esempio nel caso di $|101\rangle$ esso si può anche scrivere come $|5\rangle = |2^0 + 2^2\rangle$.

## 1.2 Gate

### 1.2.1 Gate per il singolo qbit

In questo capitolo definiamo le operazioni che ci permettono di modificare gli stati dei qbit. Le operazioni di singolo qbit sono tutte quelle operazioni che trasformano $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ in $|\psi'\rangle = \alpha'|0\rangle + \beta'|1\rangle$ con il vincolo che anche $|\psi\rangle'$ sia normalizzato. Dimostriamo che, per conservare la norme, le gate devono devono essere descritte da una matrice associata *unitaria*, tali matrici rispettano la legge $A^{-1} = A^{\dagger}$. Dato il vettore $|\psi\rangle$ normalizzato, sia A la matrice unitaria associata alla trasformazione, allora avremo:

$$|A|\psi\rangle|^2 = |A|\psi\rangle|^{\dagger}|A|\psi\rangle| = \langle\psi|A^{\dagger}A|\psi\rangle = \langle\psi|\psi\rangle = 1 \tag{1.11}$$

Il NOT (**X**), ossia la *gate* che inverte le ampiezze dei due stati, è definito nel seguete modo:

$$X : \alpha|0\rangle + \beta|1\rangle \rightarrow \alpha|1\rangle + \beta|0\rangle \tag{1.12}$$

Nella base canonica la sua azione è la seguente:

$$X\begin{bmatrix}\alpha\\ \beta\end{bmatrix} = \begin{bmatrix}\beta\\ \alpha\end{bmatrix} \tag{1.13}$$

La matrice che permette tale permutazione è la *matrice di Pauli* $\sigma_x$:

$$\sigma_x = \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix} = |1\rangle\langle 0| + |0\rangle\langle 1| \tag{1.14}$$

Come si può notare l'azione della *gate X* è lineare, questo è legato al fatto che la linearità è un proprietà fondamentale della meccanica quantistica.

Dimostriamo che date due trasformazioni unitarie la loro composizione è ancora unitaria:

$$(UV)^{\dagger} = V^{\dagger}U^{\dagger} = V^{-1}U^{-1} = (UV)^{-1} \tag{1.15}$$

tale proprietà ci permetterà di utilizzare un qualsiasi numero di gate in successione senza avere alcun problema.
Le trasformazioni di singolo qbit possono essere visualizzate come rotazioni e riflessioni rispetto agli assi x,y e z della sfera di Bloch.

In seguito descriviamo l'azione di altre due gate:

$$Z = \sigma_z = \begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix} = |1\rangle\langle 0| - |0\rangle\langle 1| \tag{1.16}$$

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} = \frac{|0\rangle + |1\rangle}{\sqrt{2}}\langle 0| + \frac{|0\rangle - |1\rangle}{\sqrt{2}}\langle 1| \tag{1.17}$$

Z lascerà inalterato lo stato $|0\rangle$ mentre cambia il segno dell'ampiezza di $|1\rangle$. Soffermiamoci su H, detta "Hadamard gate" perchè, oltre ad essere una gate utilizzata nella quasi toltalità degli algoritmi, è di più difficile comprensione non avendo un analogo classico. Applichiamo *H* allo stato $|0\rangle$:

$$\begin{aligned} H|0\rangle &= \frac{|0\rangle + |1\rangle}{\sqrt{2}} \langle 0|0\rangle + \frac{|0\rangle - |1\rangle}{\sqrt{2}} \langle 1|0\rangle \\ &= \frac{|0\rangle + |1\rangle}{\sqrt{2}} \end{aligned} \tag{1.18}$$

Avremo quindi probabilità pari a $\frac{1}{2}$ sia di misurare lo stato $|0\rangle$ che $|1\rangle$. In figura 1.2 è indicato l'effetto di H su tali stati utilizzando la sfera di Bloch. Applicare H sullo stato $|0\rangle$ significa effettuare una rotazione intorno all'asse delle y di 90° e successivamente intorno all'asse delle x di 180°. Dimostriamo che applicare 2 volte la gate H equivale ad applicare la matrice identità:

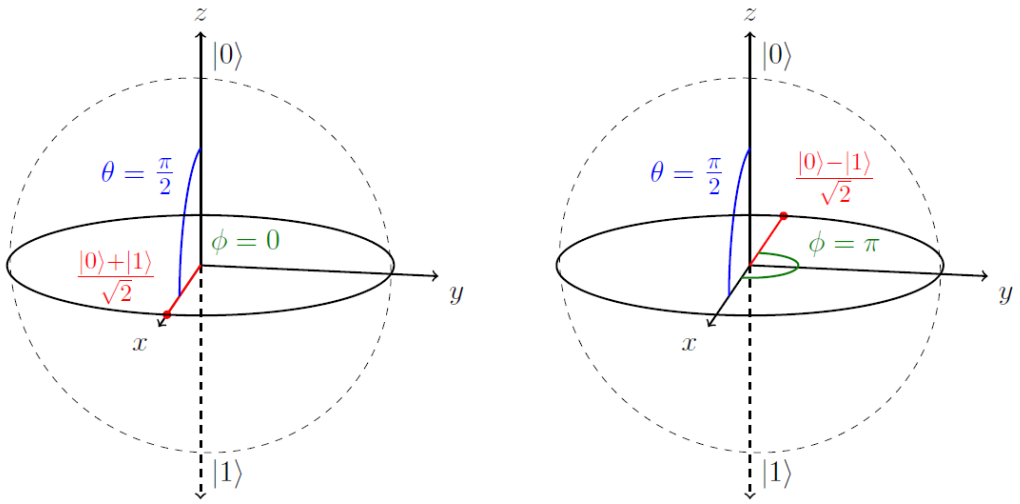
$$\left(\frac{1}{\sqrt{2}}\right)^2 \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{1.19}$$

L'effetto di tale gate è quindi quello di rendere equiprobabili tutti gli stati del sistema. Questo significa che riprendendo l'esempio dell'elettrone nei livelli energetici $|0\rangle$ e $|1\rangle$ tale gate renderà equiprobabile la presenza della particella in entrambi gli stati. In altre parole questa gate costituisce uno stato di sovrapposizione. Per analogia possiamo accomunare tale gate a un uomo che lancia una moneta, sinchè quest'ultima non si fermerà il risultato del lancio potrà essere sia testa che croce.

### 1.2.2 Gate per qbit multipli

I ragionamenti relativi alle gate di singolo qbit valgono anche per n qbit, ossia che le matrici associate devono essere unitarie. Quello che cambia è la dimensione delle matrici che, nella base canonica, dovendo lavorare in uno spazio $2^n$-dimensionale è pari a $2^n \times 2^n$. Citiamo, ad esempio, nel caso di due *qbit* la gate *CNOT* (Controlled NOT). Uno dei due qbit assumerà il ruolo di stato di controllo, mentre l'altro di stato di target. La gate valuta se lo stato di controllo è $|0\rangle$ o $|1\rangle$. Se tale stato è $|1\rangle$ inverte l'altro *qbit*, ossia lo stato di target, se incece è $|0\rangle$ non agisce. Se lo stato di controllo è il primo *qbit* le matrice è $C_{10}$, mentre se è il secondo è $C_{01}$:

$$C_{10} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}; C_{01} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{1.20}$$




(a) Basis state $|0\rangle$ (b) Basis state $|1\rangle$

Figura 1.2: effetto di H su $|0\rangle$ e $|1\rangle$

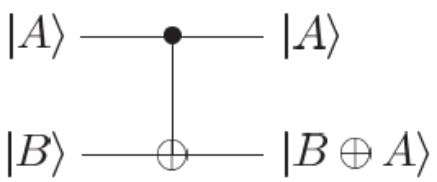



Figura 1.3: effetto di H su $|0\rangle$ e $|1\rangle$

In figura 1.3 vi è la rappresentazione circuitale di $C_{10}$, dove $\oplus$ è una somma modulo 2. Tale rappresentazione ci fa notare che il CNOT non è altro che uno XOR (exclusive OR). Lo XOR agisce nel seguete modo: dati due bit la porta restituirà 1 solo se uno solo dei due bit è 1,mentre, quando entrambi sono bassi o alti, essa restituirà zero.

Portiamo come esempio l'effetto di $C_{10}$ su $|11\rangle = (0, 0, 0, 1)$:

$$C_{10}\,|11\rangle = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = |10\rangle \tag{1.21}$$

Avendo preso $C_{10}$ lo stato di controllo è il primo, quest'ultimo essendo $|1\rangle$ permetterà alla gate di scambiare il valore del secondo qbit.

Si può dimostrare [1] il risultato di *universalità* del CNOT e delle gate a singolo qbit, ossia: *ogni gate di qbit multipli può essere compostra dalla gate CNOT e da gate di singolo qbit*. Tale risultato ci permette, quando indichiamo il grafico circuitale di una gate, di scomporre l'azione di quest'ultima con l'uso di vari CNOT e gate di singolo qbit.

## 1.3 Circuiti quantistici

Analiziamo ora un semplice circuito quantico composto da tre gate mostrato in figura 1.3, in tale figura ogni linea rappresenta un "filo" del circuito quantistico. Queste ultime potrebbero non essere dei veri fili ma rappresentare fenomeni fisici, come il passaggio di tempo, oppure lo scambio di un fotone.

Nella figura 1.3 vediamo il semplice circuito che scambia lo stato di due *qbit*, difatti le operazioni effettuate sullo $|a, b\rangle$ sono le seguenti:

$$\begin{aligned} |a, b\rangle &\rightarrow |a, a \oplus b\rangle \\ &\rightarrow |a \oplus (a \oplus b), a \oplus b\rangle = |b, a \oplus b\rangle \\ &\rightarrow |b, (a \oplus b) \oplus b\rangle = |b, a\rangle \end{aligned} \tag{1.22}$$

Il simbolo $\oplus$ indica una somma modulo due.
Esistono alcune proprietà dei circuiti classici che di solito non sono presenti in quelli quantistici: non sono permessi i *cicli* infatti i circuiti sono *aciclici*; non è implementabile il FANIN, che classicamente è l'unione dei fili utilizzata per implementare un OR, perchè, come detto prima, l'operazione non sarebbe invertibile; non è permesso il FANOUT, ossia il copiare uno stato su più qbit, perchè le leggi fondamentali della meccanica quantistica lo impediscono.
L'ultimo elemento circuitale che indichiamo in questo paragrafo indica la misura di uno stato $|\psi\rangle$, esso è rappresentato in figura 1.4. Quest'operazione, come già anticipato, teoricamente va a rompere la sovrapposizione facendo collassare lo $|\psi\rangle$ o in $|0\rangle$ o $|1\rangle$, facendo ciò lo stato assume natura classica. In figura gli stati collassati, e quindi classici, sono rappresentati dai 2 fili nella parte più a destra della figura 1.4.

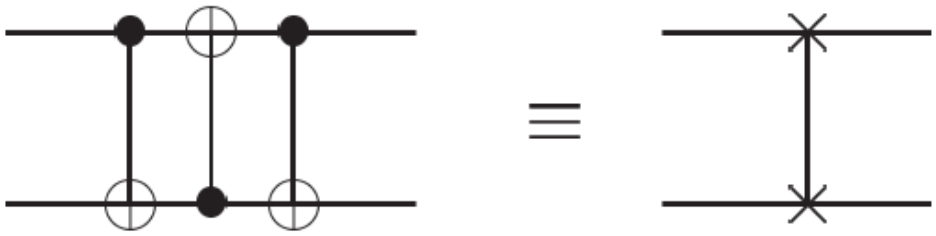


Figura 1.4: Circuito che scambia due qbit e la sua formulazione schematica

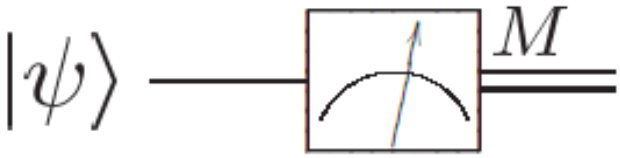



Figura 1.5: Gate che indica la misura di uno stato $|\psi\rangle$

# Capitolo 2

# Algoritmi di ricerca

Iniziamo descrivendo l'algoritmo classico per la risoluzione di un problema di ricerca di un elemento in un database disordinato. Data una lista di N elementi l'algoritmo determina, scorrendo ad uno ad uno gli elementi, se è presente l'informazione desiderata. Per fare ciò si definisce una *funzione oracolo*, inizializzata a *0*, che cambierà valore solo se verrà trovato l'elemento prescelto. La complessità computazionale di tale algoritmo è *O(N/2)*, tale approssimazione si ottiene facendo una media su tutte le posizioni in cui è possibile trovare l'elemento.

Nelle sezioni successive descriveremo gli algoritmi quantistici di searching. In particolare parleremo di uno dei primi algoritmi quantistici mai formulati, ossia quello di Grover, e un algoritmo di computazione adiabatica pensato dei fisici Jérémie Roland e Nicolas J. Cerf .

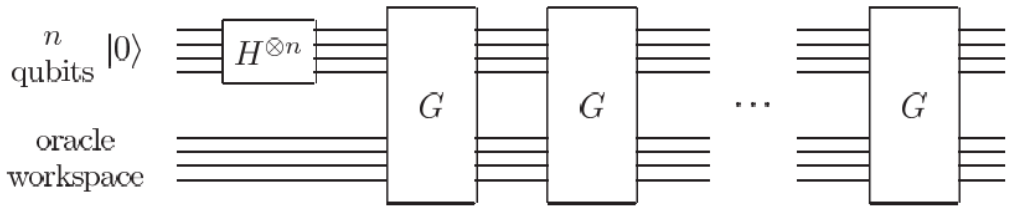
## 2.1 Algoritmo di Grover

Iniziamo associando ad ognuno degli N elementi lo stato $|i\rangle$ di un sistema a n qbit, questi ultimi generano uno spazio $2^n$ dimensionale. Possiamo fare ciò senza ledere la generalità perchè, qualora il numero degli elementi sia inferiore a $2^n$, si sceglie un set di elementi fittizi che vadano a colmare la differenza. Definiamo come funzione oracolo l'operatore unitario *O*. Sia $|w\rangle$ lo stato che si vuole trovare, possiamo rappresentare l'oracolo nel seguente modo:

$$|i\rangle \mapsto (-1)^{f(x)} |i\rangle \tag{2.1}$$

dove f(x) vale, come nel caso classico, 0 se $i \neq w$ e 1 se invece $i = w$.
Applicando tale operatore non riusciamo subito a distinguere la soluzione dai rimanenti stati, anche se lo sato di target è l'unico ad acquisire ampiezza negativa. Ciò che è stato appenna scritto è motivato dal fatto che l'oracolo modifica l'ampiezza ma non la probabilità di misurare effettivamente lo stato $|w\rangle$, essendo

Figura 2.1: Iterazione della subroutine di Grover




quest'ultima il quadrato dell'ampiezza.
Cominciamo la descrizione dell'algoritmo inizializzando lo stato a $|0\rangle$, ossia $|000...0\rangle$, ed applicando la trasformata di Hadamard ($H^{\otimes n}$). Come indicato nel capitolo relativo alle gates lo stato diventerà qundi un equa sovrapposizione degli stati $\{|i\rangle\}$:

$$|\psi_0\rangle = \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} |i\rangle \tag{2.2}$$

Se ci fermassimo a questo primo passo, andando a misurare il nostro sistema, avremo la stessa probabilità di trovare uno degli n stati. Lo scopo dell'algoritmo di Grover è quello di aumentare la probabilità di misurare lo stato target diminuendo quella dei restanti $N-1$ stati. Per fare ciò si ripete varie volte una stessa subroutine, chiamata *iterazione di Grover* G (fig. 2.1), composta da 4 passi:

1. Applicare l'oracolo O
2. Applicare la trasformata di Hadamard $H^{\otimes n}$
3. Cambiare il segno dell'ampiezza degli stati che differiscono da $|w\rangle$
4. Applicare la trasformata di Hadamard $H^{\otimes n}$

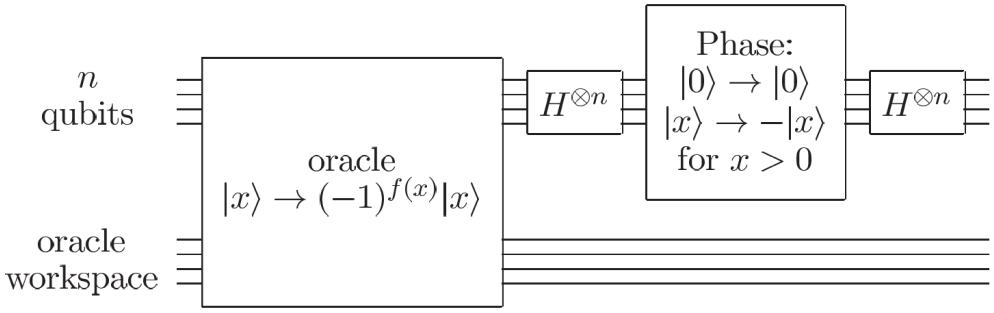
L'algoritmo associato a tale subroutine è indicato nella figura 2.2 (in tale figura $|w\rangle$ è stato scelto come $|0\rangle$). Utilizziamo la seguente notazione sugli stati del nostro sistema: posto $|\psi_0\rangle$ lo stato iniziale, chiameremo $|\psi_i\rangle$ lo stato generato applicando *i* volte la subroutine di Grover.

Definiamo l'operatore oracolo nel seguente modo:

$$U_w = 1 - 2\,|w\rangle\,\langle w| \tag{2.3}$$

La sua azione è analoga a quella in Eq 2.1, ossia $U_w$ cambia il segno dell'ampiezza del solo stato target. Applicando $U_w$ allo stato iniziale $|\psi_0\rangle$ avremo:

Figura 2.2: Iterazione di Grover




$$\begin{aligned} U_w |\psi_0\rangle &= (1 - 2 |w\rangle \langle w|) |\psi_0\rangle \\ &= \frac{1}{\sqrt{N}} \sum_{i=0, i\neq w}^{N-1} |i\rangle - \frac{1}{\sqrt{N}} |w\rangle \end{aligned} \tag{2.4}$$

$U_w$ è quindi una buona funzione oracolo perchè ha invertito il segno dell'ampiezza solo dello stato target.
Per i passi 2 e 4 serve applicare semplicemente la trasformata di Hadamard. Per il passo 3 si sceglie invece l'operatore:

$$U_t = 2 |w\rangle \langle w| - I \tag{2.5}$$

Valutiamone l'azione sullo stato $|\psi_0\rangle$ per verificare l'effettivo cambio di fase di tutti gli stati tranne che di $|w\rangle$:

$$\begin{aligned} U_t |\psi_0\rangle = (2 |w\rangle \langle w| - I) |\psi_0\rangle &= \frac{2}{\sqrt{N}} |w\rangle - \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} |i\rangle \\ = \frac{1}{\sqrt{N}} |w\rangle - \frac{1}{\sqrt{N}} \sum_{i=0, i\neq w}^{N-1} |i\rangle & \end{aligned} \tag{2.6}$$

L'operatore finale associata agli step 2, 3 e 4 della subroutine di Grover sarà quindi:

$$\begin{aligned} U_s &= H^{\otimes n}(2 |w\rangle \langle w| - I)H^{\otimes n} \\ &= 2H^{\otimes n} |w\rangle \langle w| H^{\otimes n} - H^{\otimes n}H^{\otimes n} \\ &= 2 |\psi_0\rangle \langle \psi_0| - 1 \end{aligned} \tag{2.7}$$

Di seguito studiamo l'effetto di una iterazione di G:

$$\begin{aligned}|\psi_1\rangle = G\,|\psi_0\rangle = U_s U_w\,|\psi_0\rangle &= (2\,|\psi_0\rangle\,|\psi_0\rangle - 1)(1 - 2\,|w\rangle\,\langle w|)\,|\psi_0\rangle \\ &= 2\,\langle\psi_0|\psi_0\rangle\,|\psi_0\rangle - |\psi_0\rangle - 4\,\langle w|\,|\psi_0\rangle\rangle\,\langle|\psi_0\rangle\,|w\rangle\,|\psi_0\rangle + 2\,\langle w|\psi_0\rangle\,|w\rangle \\ &= |\psi_0\rangle - 4|\langle w|\psi_0\rangle|^2\,|\psi_0\rangle + 2\,\langle w|\psi_0\rangle\,|w\rangle \\ &= |\psi_0\rangle - \frac{4}{N}\,|\psi_0\rangle + \frac{2}{\sqrt{N}}\,|w\rangle\end{aligned} \tag{2.8}$$

Dove al posto di $\langle w|\psi_0\rangle$ abbiamo sostituito il valore $\frac{1}{\sqrt{N}}$. Si vede che l'effetto di G è quello di aumentare l'ampiezza di $|w\rangle$ e diminuire le restanti. Infatti, essendo $\frac{2}{\sqrt{N}} > \frac{4}{N}$, è facile notare che l'ampiezza in $|\psi_1\rangle$ di $|w\rangle$ è maggiore rispetto a quella degli altri stati. Infatti le ampiezze di $|i\rangle$, con $i \neq w$, diminuisce di $\frac{4}{N}$ mentre l'ampiezza di $|w\rangle$ aumenta di $\frac{2}{\sqrt{N}} - \frac{4}{n}$.

Un metodo efficace per analizzare il processo è quello mostrato nella figura 2.3. Poniamo $|w\rangle = |\beta\rangle$ e chiamiamo $|\alpha\rangle = |w^{\perp}\rangle = |\psi_i\rangle - \langle\psi_i|w\rangle\,|w\rangle$ lo stato ortogonale al nostro target. Disegnamo un grafico cartesiano dove gli assi sono $|\alpha\rangle$ e $|\beta\rangle$, in questo rappresentiamo il vettore $|\psi_i\rangle$ (che in figura chiamiamo semplicemente $|\psi\rangle$) con origine nel centro degli assi e chiamiamo $\gamma$ l'angolo compreso tra $|\psi_i\rangle$ e $|\alpha\rangle$. Poniamo $\langle w|\psi_0\rangle = sen\frac{\theta}{2} = \frac{1}{\sqrt{N}}$ dove, per n molto grande, possiamo espandere il seno e porre $\frac{\theta}{2}$ uguale a $\frac{1}{\sqrt{N}}$. Applicando G $\gamma$ aumenterà, con esso quindi crescerà il prodotto scalare $\langle w|\psi_i\rangle$ e ciò porterà all' aumento della probabilità di misurare lo stato $|w\rangle$. Come si vede sempre nella figura 2.3 l'operatore O applica una riflessione di $|\psi_i\rangle$ rispetto all'asse $|\alpha\rangle$, mentre l'operatore G ribalterà nuovamente $|\psi_i\rangle$ e aumenterà l'angolo $\gamma$ .

Valutiamo l'incremento che si ha ad ogni applicazione della subroutine di Grover riprendendo il risultato dell' equazione 2.8. L'incremento dato dall' applicazione di G è pari a:

$$\frac{2}{\sqrt{N}} - \frac{4}{N} = \frac{2N - 4\sqrt{N}}{N\sqrt{N}} \approx \frac{2}{\sqrt{N}} \tag{2.9}$$

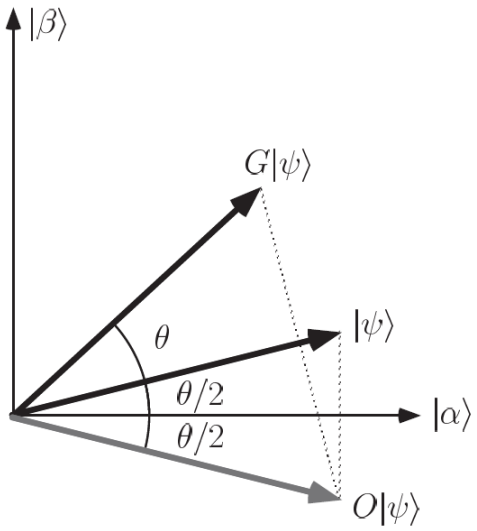
ossia, essendo $\frac{\theta}{2} = \frac{1}{\sqrt{N}}$, $\gamma$ aumenterà di un angolo $\theta$. Questo significa che applicando G varie volte avremo che $\gamma$ da $\frac{\theta}{2}$ diventerà $\frac{3\theta}{2}$, poi $2\theta$ e così via.

Il numero di step necessari per avere un risultato quasi certo, cioè necessario a "ruotare $\psi_0$ su $|w\rangle$, sarà quindi :

$$N_{step}\theta \approx \frac{\pi}{2} \rightarrow N_{step} \approx \frac{\pi}{4}\sqrt{N} \tag{2.10}$$

La 2.10 indica il grande pregio di questo algoritmo quantistico,ossia che la complessità computazionale è pari a O($\sqrt{N}$). Quest'ultima è di gran lunga più bassa di quella della programmazione classica, che ricordiamo essere O(N/2).

Figura 2.3: rappresentazione grafica




### 2.1.1 Ricerche multiple

Supponiamo di voler cercare M elementi. I due sottospazi generati inizialmente, ossia quello delle soluzioni, che prima era composto dal solo vettore $|w\rangle$, e quello ortogonale agli stati target, saranno generati da:

$$|\alpha\rangle = \frac{1}{\sqrt{N-M}} \sum_{x}{}'' |x\rangle \tag{2.11}$$

$$|\beta\rangle = \frac{1}{\sqrt{M}} \sum_{x}{}' |x\rangle \tag{2.12}$$

con $\sum_x'$ la somma sugli stati target e con $\sum_x''$ la somma sui restanti.
Lo stato del sistema sarà quindi:

$$\begin{aligned} |\psi\rangle &= \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle \\ &= \frac{1}{\sqrt{N}} \sum_{x}{}'' |x\rangle + \frac{1}{\sqrt{N}} \sum_{x}{}' |x\rangle \\ &= \sqrt{\frac{N-M}{N}} |\alpha\rangle + \sqrt{\frac{M}{N}} |\beta\rangle \end{aligned} \tag{2.13}$$

Posto $\cos\frac{\theta}{2} = \sqrt{\frac{N-M}{N}}$ la 2.13 diventa:

$$|\psi\rangle = \cos\frac{\theta}{2} |\alpha\rangle + \sin\frac{\theta}{2} |\beta\rangle \tag{2.14}$$

Applichiamo ora k volte la subroutine G allo stato $|\psi\rangle$:

$$|\psi\rangle = \cos\frac{(2k+1)\theta}{2}|\alpha\rangle + \sin\frac{(2k+1)\theta}{2}|\beta\rangle \tag{2.15}$$

Ad ogni iterazione avremo quindi che l'angolo tra $|\psi\rangle$ e $|\alpha\rangle$ aumenterà di $\theta$, come nel caso precedente.

Infine valutiamo quante iterazioni sono necessarie per avere un risultato quasi certo. Per far coincidere quasi del tutto lo stato su $|\beta\rangle$ necessitiamo di una rotazione pari a $\arccos\sqrt{\frac{M}{N}}$, il numero di iterazioni della subroutine G necessarie è:

$$R = CI\Big(\frac{\arccos\sqrt{M/N}}{\theta}\Big) \tag{2.16}$$

con CI una funzione che approssima all'intero più vicino.
Se supponiamo $M \leq \frac{N}{2}$ segue che:

$$\frac{\theta}{2} \geq \sin\frac{\theta}{2} = \sqrt{\frac{M}{N}} \tag{2.17}$$

Notiamo dalla 2.16 che, essendo $\arccos\sqrt{\frac{M}{N}} \leq \frac{\pi}{2}$, si ha $R \leq \pi/2\theta$, sostituendo in questa ultima disuguaglianza anche la 2.17 avremo:

$$R \leq \frac{\pi}{4}\sqrt{\frac{N}{M}} \tag{2.18}$$

ossia l'ordine di grandezza del numero di iterazioni necessarie è pari a $O(\sqrt{N/M})$.

### 2.1.2 Simulazione numerica

Portiamo come esempio, per esplicitare un circuito quantistico, il caso di un algoritmo di ricerca per 2 *qbit* (fig. 2.4). Essendo la dimesione del nostro spazio pari a N=4 la funzione oracolo sarà $f(x_0) = 1$, con $x_0$ l'elemento oggetto della nostra ricerca, e $f(x) = 0$ per $x \neq x_0$. La schematizzazione di tale funzione è indicata in figura 2.5, dove le prime due linee indicano il valore di $x$ e la terza linea il valore della funzione oracolo.

Inizializiamo lo stato nella sovrapposizione uniforme $|\psi\rangle = (|00\rangle + |01\rangle + |10\rangle + |11\rangle)/2$, avremo quindi che $\gamma$, ossia l'angolo compreso tra lo stato del nostro sistema e l'asse $\alpha$ (vedi fig. 2.3), sarà:

$$\sin\gamma = 1/2 \rightarrow \gamma = 30^\circ \tag{2.19}$$

La probabilità di misurare lo stato $x_0$ al passo 0 è quindi $P_0 = \frac{1}{4}$. Utilizzando la notazione precendente, al passo zero $\gamma = \theta/2$ segue che $\theta = 60^\circ$. Quest'ultimo è

Figura 2.4: Circuito completo

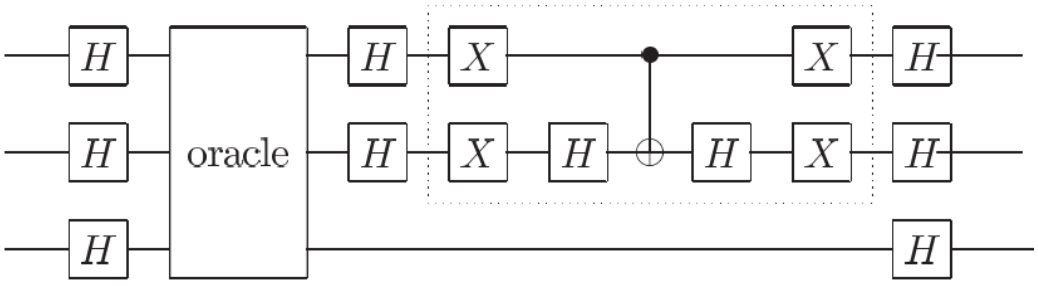
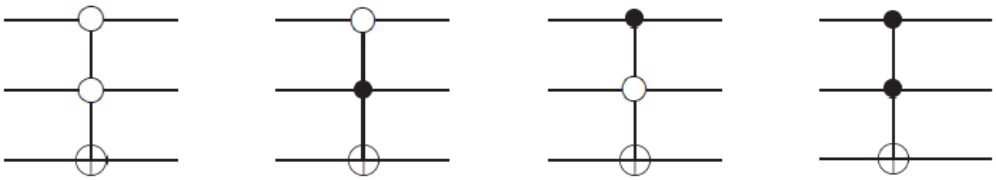



Figura 2.5: Quattro circuiti per implementare la funzione oracolo per la ricerca rispettivamente di $|00\rangle$,$|01\rangle$,$|10\rangle$,$|11\rangle$



l'angolo di rotazione dato dall'applicazione della subroutine di Grover, ciò comporta quindi che $|\psi_1\rangle$ coincide con l'asse $|\beta\rangle$. Quest'esempio ci permette anche di comprendere la velocità dell'algoritmo. In un passaggio, a differenza dei 2 in media del caso classico, riusciamo a trovare con estattezza un elemento in 4 posti disponibili. Supponiamo che $|x_0\rangle = |00\rangle$. Visualizziamo su di un grafico come vengono modificate le ampiezze applicando la subroutine di Grover. In figura 2.6 è rappresentato lo stato iniziale (Eq. 2.2), applicando a questo la funzione oracolo l'ampiezza di $|00\rangle$ diventerà negativa (fig 2.7). Infine dopo l'azione si $U_w$ e $U_s$ lo stato $|00\rangle$ avrà ampiezza aumentata, mentre i restati elementi della base avranno ampiezza diminuita (fig 2.8). Per tale effetto l'algoritmo di Grover viene anche chiamato di "Amplitude amplification".

.

Figura 2.6: $|\psi\rangle$ normalizzato

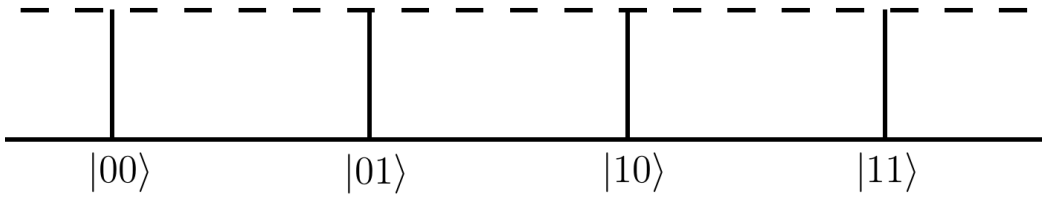


Figura 2.7: Effetto di O su $|\psi\rangle$

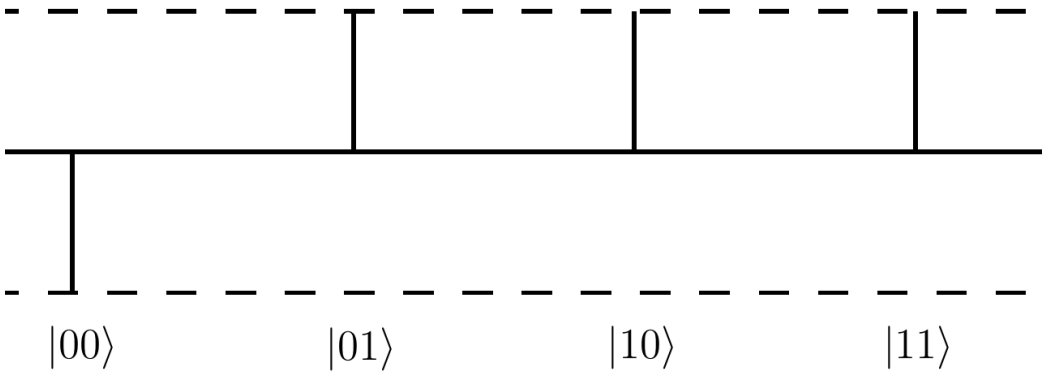



Figura 2.8: Effetto di G su $|\psi\rangle$

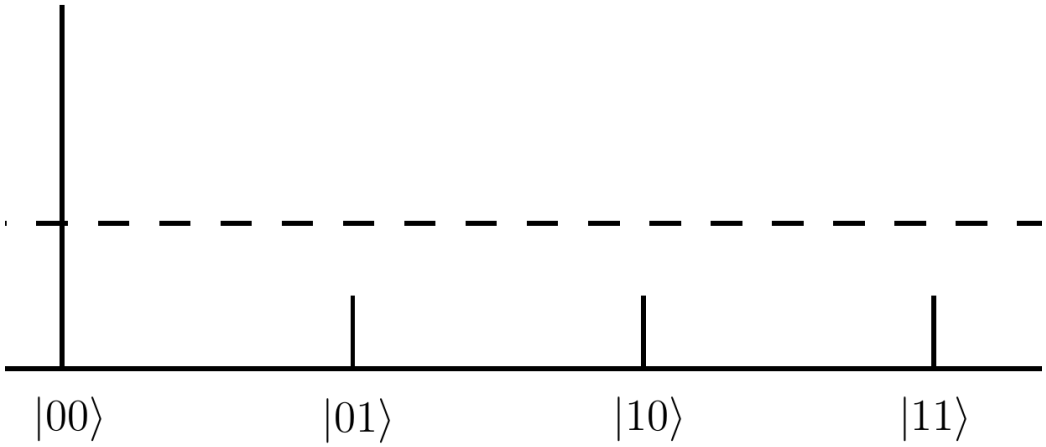


Figura 2.9: Rappresentazione delle ampiezze di $\psi_0$


L'ultima parte di questo capitolo è dedicata a una simulazione fatta su Wolfram Mathematica (appendice 1) dell'algoritmo studiato. In tal caso abbiamo preso in considerazione uno spazio generato da 10 qbit ossia di dimensione $2^{10} = 1024$ e come target è stato scelto lo stato $|134\rangle$. Attraverso un ciclo DO sono stati applicati $\frac{\pi}{4}\sqrt{N} \simeq 25$ volte gli operatori prima studiati. Si riportano di seguito due tipologie di grafici relativi alla simulazione, una mostra le ampiezze degli stati passo dopo passo (come in figura 2.8) mentre l'altra è la rappresentazione sul piano bidimensionale dello stato $\psi$ (come in figura 2.3) . In fig. 2.9 e 2.10 è mostrato lo stato iniziale $\psi_0$, mentre nelle figure 2.11 e 2.12 sono rappresentate le grandezze relative al terzo passo. Infine le figure 2.13 e 2.14 riguardano il 10 passo e le figure 2.15 e 2.16 lo stato finale. Nella 2.16 si nota la quasi totale rotazione sullo stato target a meno di un piccolo angolo, quest'ultimo è legato alla probabilità di misurare uno stato diverso da quello target.

Figura 2.10: Rappresentazione di $\psi_0$

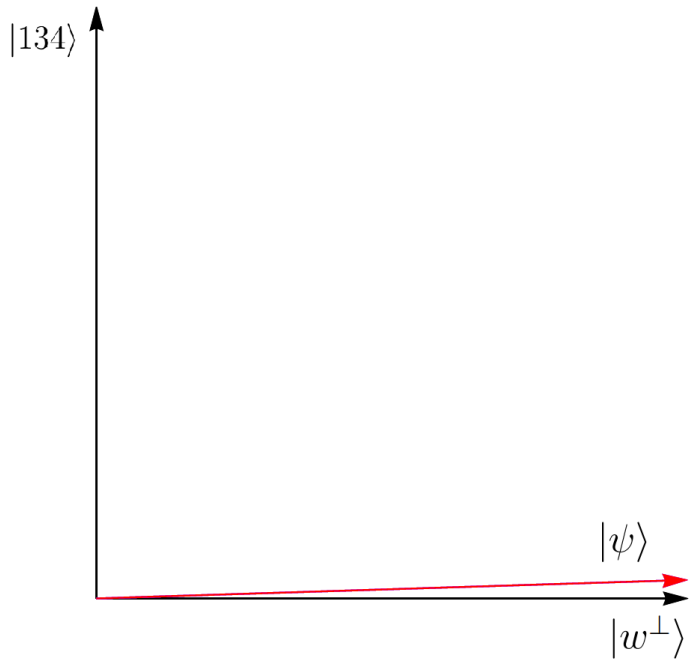



Figura 2.11: Rappresentazione delle ampiezze di $\psi_3$

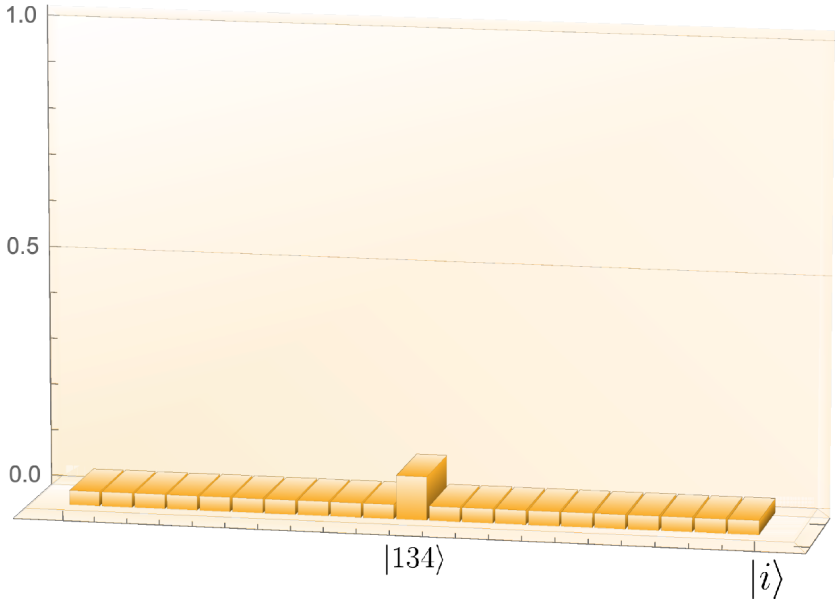


Figura 2.12: Rappresentazione di $\psi_3$

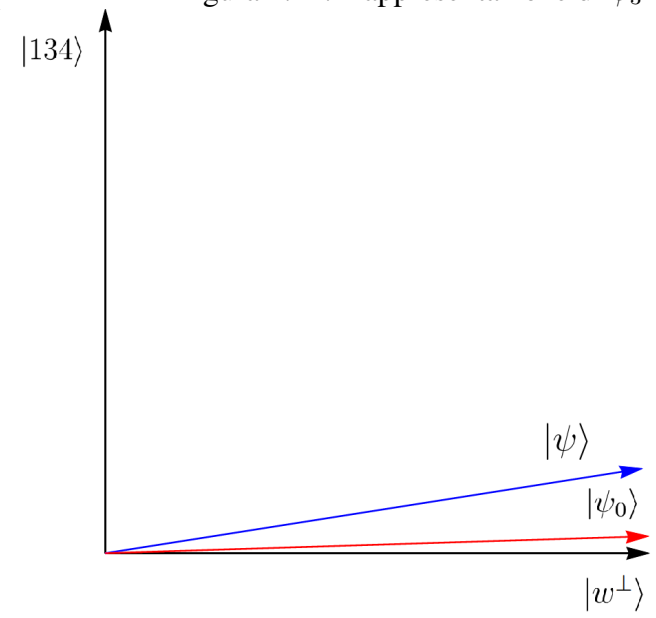



Figura 2.13: Rappresentazione delle ampiezze di $\psi_1 0$

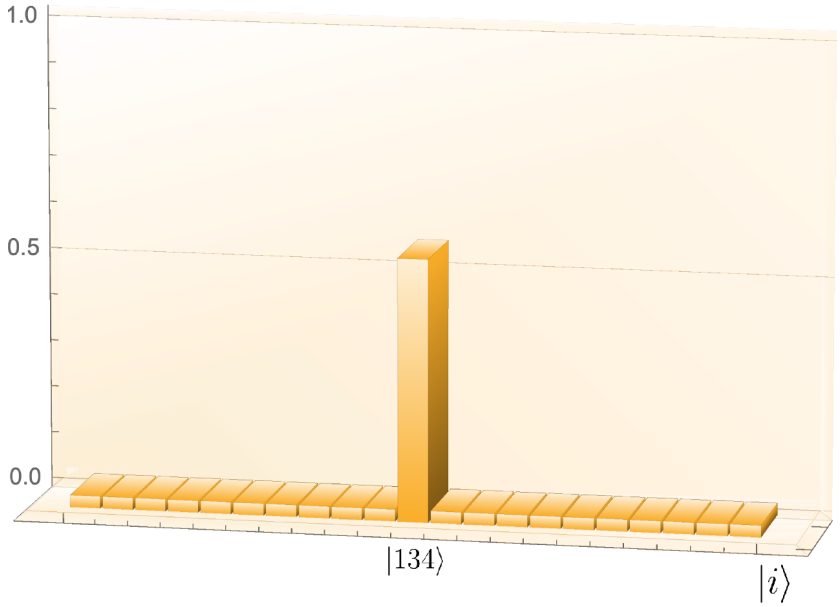


Figura 2.14: Rappresentazione di $\psi_1 0$

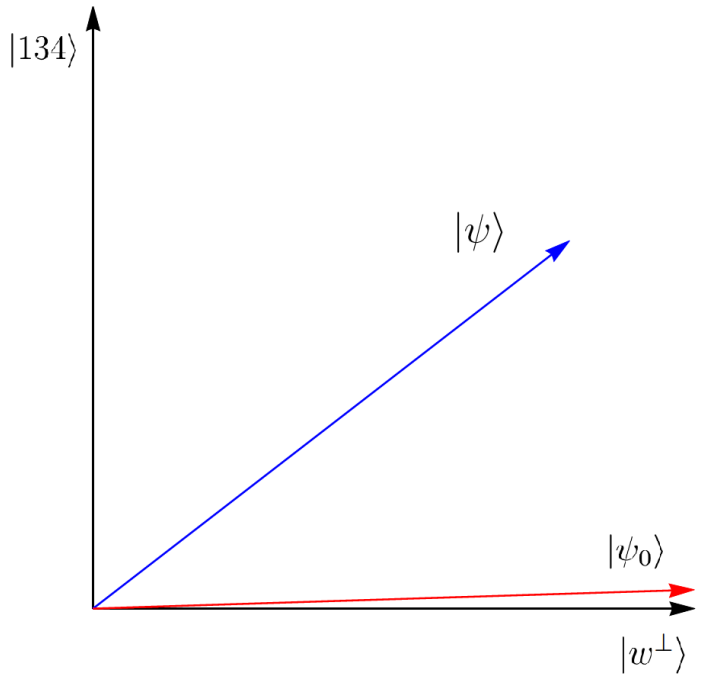



Figura 2.15: Rappresentazione delle ampiezze di $\psi_2 5$

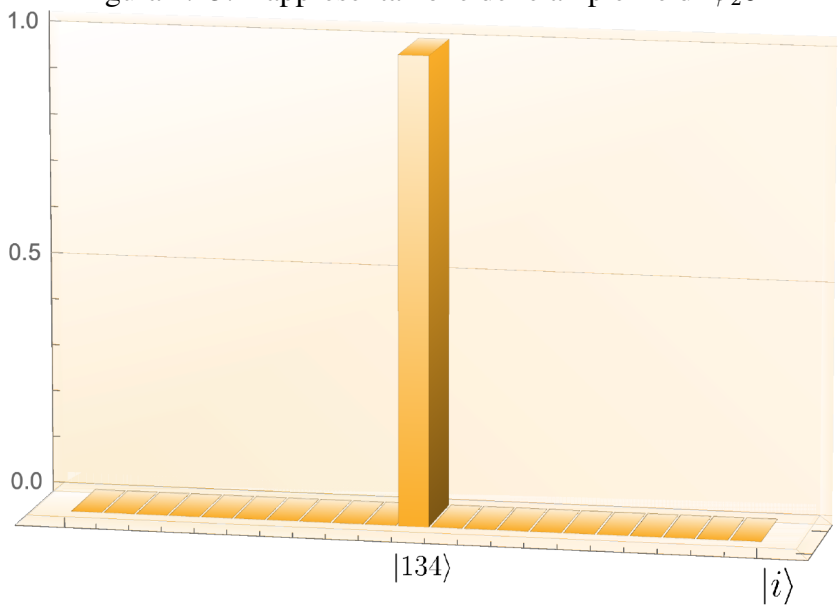
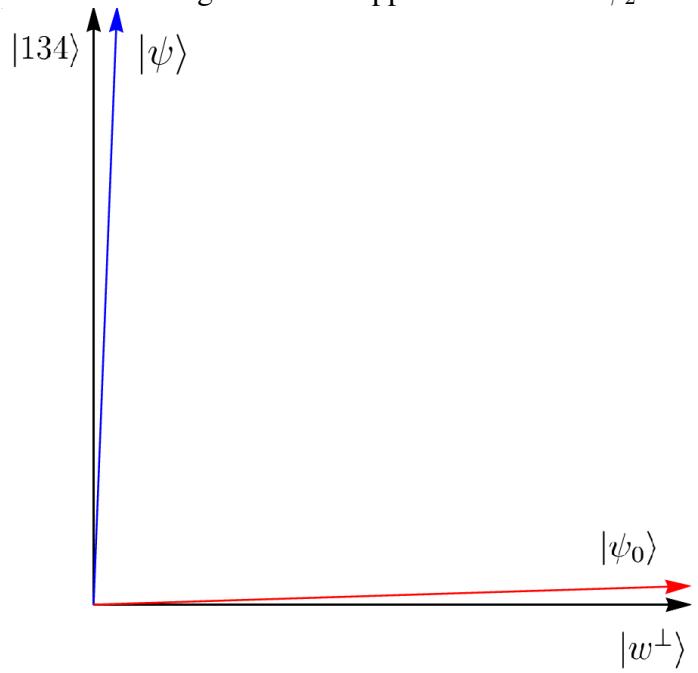


Figura 2.16: Rappresentazione di $\psi_2 5$



## 2.2 Algoritmo adiabatico di Roland e Cerf

La computazione quantistica adiabatica è un approccio diverso alla programmazione di algoritmi quantistici. Il teorema su cui si basa questa branca è il teorema adiabatico, esso afferma che:

**Theorem 1.** *Un sistema fisico $|\psi\rangle$, che evolve nel tempo seguendo l'equazione di Schrödinger $i\hbar\frac{d}{dt}|\psi(t)\rangle = H(t)|\psi(t)\rangle$, rimane in un suo autostatoto istantaneo se si applica ad esso una perturbazione sufficientemente lenta rispetto al gap energetico tra l'autostato e il restante spettro dell'hamiltoniana.*

Per "sufficientemente lenta" si intende che il rate di evoluzione deve essere più basso di un tempo caratteristico. Quest'ultimo dipende inversamente dalla minima gap tra i livelli energetci, che indichiamo come:

$$g_{min} = min_{0\leq t\leq T}[E_1(t) - E_0(t)] \tag{2.20}$$

Con T il tempo totale di evoluzione e $E_n$ l'autovalore n-esimo dell'hamiltoniana al tempo t. Chiamiamo l'elemento massimo della matrice $\frac{dH}{dt}$ :

$$D_{max} = max_{0\leq t\leq T}\left|\left\langle \frac{dH}{dt} \right\rangle_{1,0}\right| \tag{2.21}$$

Il nostro scopo è che, dopo il tempo di evoluzione T, lo stato si trovi con quasi assoluta certezza nello stato fondamentale H(t). Quest'ultima frase la si può tradurre matematicamente con l'espressione seguente:

$$|\langle E_0; T|\psi(T)\rangle|^2 \geq 1 - \epsilon^2 \tag{2.22}$$

La 2.22 sta ad indicare che la probabilità di misurare lo stato $|\psi\rangle$, al tempo T, e trovare lo stato fondamentale dell'hamiltoniana, $|E_0; T\rangle$, deve essere praticamente 1. Il valore associato alla 2.22 è anche chiamato *fidelity*. Quell' $\epsilon$ sta ad indicare l'accuratezza con cui vogliamo effettuare la misura. Si può provare [3] che :

$$\frac{D_{max}}{{g_{min}}^2} \leq \epsilon \tag{2.23}$$

dove $\epsilon \ll 1$. Intuitivamente la 2.23 è spiegabbile dicendo che la "velocità" con cui varia l'hamiltoniana deve essere molto più piccola della gap tra gli stati così che non permettere le eccitazioni dette di Landau-Zener.

Iniziamo ora con la discussione dell'algoritmo di searching adiabatico. Siano $|i\rangle$, con i che va da 0 a $N-1$, gli N vettori che compongono la base del nostro

spazio e sia $|m\rangle$ l'obiettivo del nostro algoritmo. Come stato iniziale scegliamo la sovrapposizione unifome degli elementi della base:

$$|\psi_o\rangle = \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} |i\rangle \tag{2.24}$$

Scriviamo ora un hamiltoniana interagente che al tempo t=T sia $H(T) = H_m$, ossia un hamiltoniana che ha come stato fondamentale lo stato target, e che al tempo t=0 è $H(0) = H_0$, ossia un' hamiltoniana semplice non interagente. La nostra scelta ricade sui seguenti operatori $H_m = 1 - |m\rangle \langle m|$ e
$H_0 = 1 - |\psi_o\rangle \langle \psi_o|$.

Come hamiltoniana finale dipendente dal tempo scegliamo la combinazione di $H_m$ e $H_0$:

$$H(t) = [1 - s(t)]H_0 + s(t)H_m \tag{2.25}$$

con s(t) una funzione monotona che deve soddisfare le condizioni s(0)=0 e s(T)=1, con queste condizioni, per t=0, avremo $H(0) = H_0$ e, per t=T, $H(T) = H_m$.

Di seguito si fa notare che tutti gli elementi diagonali della matrice H(t) sono uguali $\forall i \neq m$ :

$$\begin{aligned} \langle i|H(t)|i\rangle &= \langle i| \left\{[1 - s(t)](1 - |\psi_o\rangle \langle \psi_o|) + s(t)(1 - |m\rangle \langle m|)\right\} |i\rangle \\ &= [1 - s(t)] \langle i| (1 - |\psi_o\rangle \langle \psi_o|) |i\rangle + s(t) \langle i|i\rangle \\ &= [1 - s(t)] \left(\langle i|i\rangle - \frac{1}{N}\right) + s(t) = [1 - s(t)] \left(1 - \frac{1}{N}\right) + s(t) \end{aligned} \tag{2.26}$$

Questo ci permette di ridurre il problema ad uno spazio generato da $|m\rangle$ e il suo complemento ortogonale. Lo stato iniziale può essere scritto nel seguente modo:

$$\begin{aligned} |\psi_o\rangle &= \frac{1}{\sqrt{N}} |m\rangle + \frac{1}{\sqrt{N}} \sum_{i=0, i\neq m}^{N-1} \frac{\sqrt{N-1}}{\sqrt{N-1}} |i\rangle \\ &= \frac{1}{\sqrt{N}} |m\rangle + \sqrt{\frac{N-1}{N}} |m^{\perp}\rangle \end{aligned} \tag{2.27}$$

Essendo la nuova base composta da 2 elementi $|m\rangle, |m^{\perp}\rangle$ possiamo studiare la

matrice associata ad $H(t)$ come una matrice 2x2 composta dai seguenti elementi:

$$\begin{aligned}
H_{mm} &= \langle m| H(t) |m\rangle = \langle m| \{[1-s(t)](1-|\psi_o\rangle \langle\psi_o|) + s(t)(1-|m\rangle \langle m|)\} |m\rangle \\
&= [1-s(t)](1-\frac{1}{N}) + s(t)(1-1) = (1-s)\frac{N-1}{N} \\
H_{m^\perp m^\perp} &= \langle m| H(t) |m\rangle = \langle m^\perp| \{[1-s(t)](1-|\psi_o\rangle \langle\psi_o|) + s(t)(1-|m\rangle \langle m|)\} |m^\perp\rangle \\
&= [1-s(t)](1-\frac{N-1}{N}) + s(t) = \frac{1-s}{N} + s \\
H_{m^\perp m} &= H_{mm^\perp} = \langle m| H(t) |m^\perp\rangle = \langle m| \{[1-s(t)](1-|\psi_o\rangle \langle\psi_o|) + s(t)(1-|m\rangle \langle m|)\} |m^\perp\rangle \\
&= -[1-s(t)]\frac{\sqrt{N-1}}{N}
\end{aligned} \tag{2.28}$$

Quindi la matrice assumerà la seguente forma:

$$H(t) = \begin{bmatrix} (1-s)\frac{N-1}{N} & -(1-s)\frac{\sqrt{N-1}}{N} \\ -(1-s)\frac{\sqrt{N-1}}{N} & \frac{1-s}{N} + s \end{bmatrix} \tag{2.29}$$

Risolvendo il problema agli autovalori della matrice H(t) troviamo i livelli energetici associati agli autostati istantanei:

$$det(H(t)-\lambda\mathbb{1}) = \Big[(1-s)\frac{N-1}{N} - \lambda\Big]\Big[\frac{1-s}{N} + s - \lambda\Big] - \Big[(1-s)\frac{\sqrt{N-1}}{N}\Big]^2 = 0 \tag{2.30}$$

Le soluzioni della 2.30 sono $\lambda = \frac{1 \pm \sqrt{1-4s(1-s)\frac{N-1}{N}}}{2}$. Come si vede dalla figura 2.6 oltre i due autovalori prima determinati, indicati con il blu e l'arancione, ve ne saranno altri N-2 tutti degeneri in 1 che non influiranno nell'algoritmo. Definiamo la distanza tra i due livelli energetici come $g(s)$ (fig. 2.10):

$$g(s) = \sqrt{1 - 4s(1-s)\frac{N-1}{N}} \tag{2.31}$$

Il minimo di tale funzione è $g_{min} = \frac{1}{\sqrt{N}}$ per $s = 1/2$

Figura 2.17: Autovalori dell'hamiltoniana $H(t)$

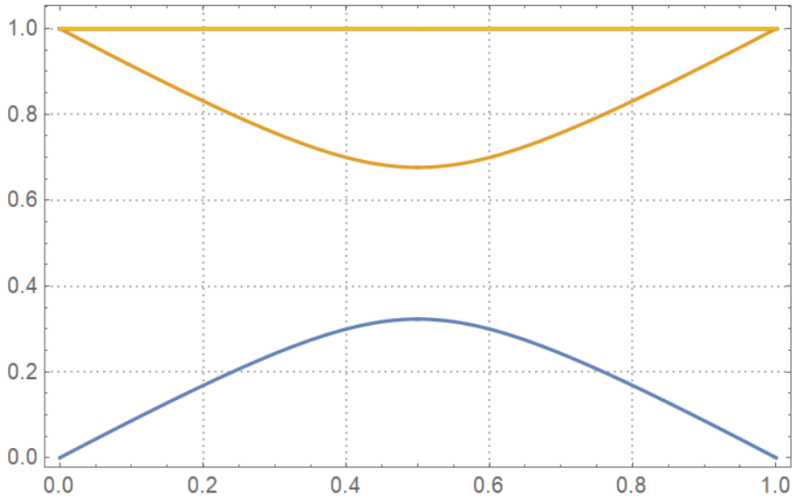
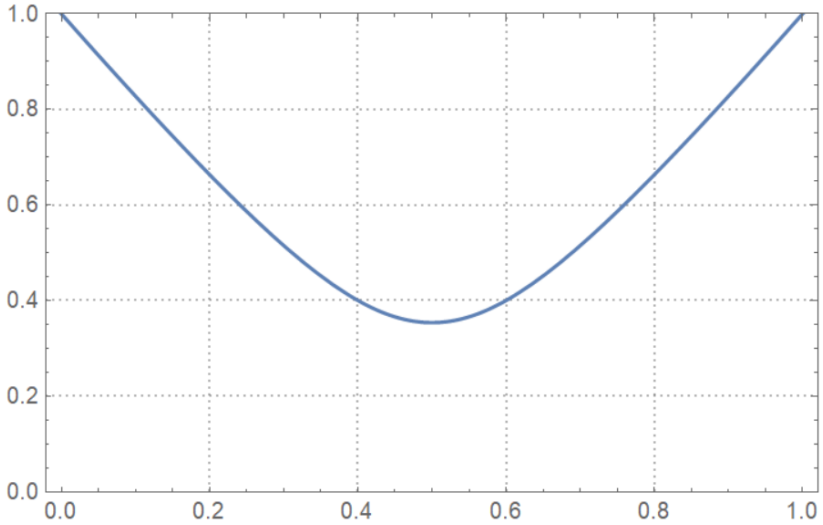



Figura 2.18: Grafico di $\Delta\lambda$



Ora è arrivato il momento di scegliere la dinamica dell'evoluzione del nostro stato, ossia scegliere la funzione di annealing s(t). La scelta più semplice che si può compiere è la *schedule lineare* ponendo $s(t) = t/T$. Data la precedente posizione, s(t) è biunivoca e possiamo esplicitare H per s. Questo comporta che la derivata dell'operatore sarà:

$$\left\langle \frac{dH}{dt} \right\rangle_{1,0} = \frac{ds}{dt} \left\langle \frac{d\tilde{H}}{ds} \right\rangle_{1,0} = \frac{1}{T} \left\langle \frac{d\tilde{H}}{ds} \right\rangle_{1,0} \tag{2.32}$$

con $\tilde{H} = H_m - H_0$.
Infine essendo:

$$\left| \left\langle \frac{d\tilde{H}}{ds} \right\rangle_{1,0} \right| \leq 1 \tag{2.33}$$

Sostituendo la 2.33 nella 2.32 la condizione di adiabaticità 2.23 diventa:

$$\frac{1}{T g_{min}^2} \leq \epsilon \rightarrow T \geq \frac{N}{\epsilon} \tag{2.34}$$

La disuguaglianza 2.34 indica che il tempo di *annealing*, e quindi la complessità computazionale, è O(N) come nel caso classico.

Per aumentare la velocità di convergenza bisogna imporre la condizione di adiabaticità localmente, e non più globalmente come del caso della shedule lineare, scegliendo così un evoluzione più veloce, dove il gap è grande, e più lenta, quando il gap si restringe. La condizione di locale adiabaticità è la seguente:

$$\left| \frac{ds}{dt} \right| \leq \frac{g^2(s)}{|\langle \frac{dH}{ds} \rangle_{1,0}|} \epsilon \tag{2.35}$$

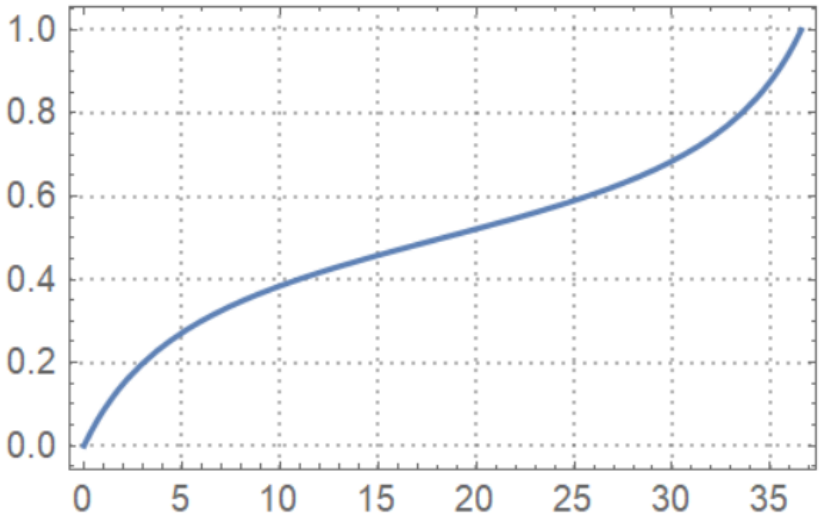
Per la posizione 2.33 poniamo $|\langle \frac{d\tilde{H}}{ds} \rangle_{1,0}| = 1$, e determiniamo s(t) risolvendo la seguente equazione differenziale con le condizioni al contorno:

$$\begin{cases} |\frac{ds}{dt}| = \epsilon g^2(s) = \epsilon(1 - 4s(1-s)\frac{N-1}{N}) \\ s(0) = 0 \\ s(T) = 1 \end{cases} \tag{2.36}$$

Tale equazione è facilmente risolvibile essendo a variabili separabili:

$$\begin{aligned} &\frac{ds}{dt} = \epsilon(1 - 4s(1-s)\frac{N-1}{N}) \\ &\frac{ds}{\epsilon\frac{1}{N}[1 + (2s-1)^2(N-1)]} = dt \\ &t(s) = \frac{1}{2\epsilon}\frac{N}{\sqrt{N-1}} \arctan[\sqrt{N-1}(2s-1)] + C \end{aligned} \tag{2.37}$$

Figura 2.19: Grafico di t(s)




Per determinare C utilizziamo una delle due condizioni al contorno:

$$\begin{aligned} t(0) = 0 &= \frac{1}{2\epsilon}\frac{N}{\sqrt{N-1}}\arctan[\sqrt{N-1}(0-1)] + C \\ C &= \frac{1}{2\epsilon}\frac{N}{\sqrt{N-1}}\arctan\sqrt{N-1} \end{aligned} \tag{2.38}$$

Andando a sostituire la 2.38 nella 2.37 avremo infine:

$$t(s) = \frac{1}{2\epsilon}\frac{N}{\sqrt{N-1}}\{\arctan[\sqrt{N-1}(2s-1)] + \arctan(\sqrt{N-1})\} \tag{2.39}$$

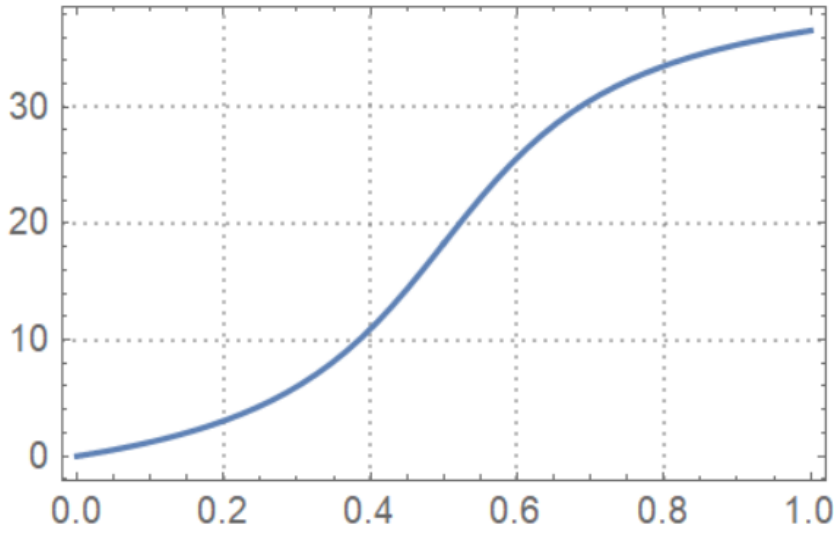
rappresentata in figura 2.11.

Esplicitando s dalla 2.39 deduciamo la schedule:

$$s(t) = \frac{1}{2}\Big[1 + \frac{\tan\frac{2\sqrt{N-1}t\epsilon - N\arctan\sqrt{N-1}}{N}}{\sqrt{N-1}}\Big] \tag{2.40}$$

s(t) è rappresentata nella figura 2.12.

Figura 2.20: Grafico di s(t)



Determiniamo infine il tempo di annealing per questo caso. Ponendo T=1 nella 2.39 avremo:

$$T = t(1) = \frac{1}{\epsilon}\frac{N}{\sqrt{N-1}} arctan\sqrt{N-1} \tag{2.41}$$

Per N tendente all'infinito $\arctan\sqrt{N-1} \approx \frac{\pi}{2}$, segue che:

$$T \approx \frac{\pi\sqrt{N}}{2\epsilon} = o(\sqrt{N}) \tag{2.42}$$

ossia il risultato cercato.

### 2.2.1 Simulazione numerica

In appendice B è riportato un algoritmo creato con Wolfram Mathematica che simula l'algoritmo di searching adiabatico.

In questo algoritmo sono stati presi in cosiderazione un caso a 3 qbit e uno a 10 qbit. Gli spazi generati da questi avranno rispettivamente una dimensione di $2^3 = 8$ e di $2^{10} = 1024$. Si è poi definita l'hamiltoniana interagente H(t) e, per calcolare gli autostati istantanei di quest'ultima, si è suddiviso l'intervallo temporane in mille parti. In ognuna di queste sezioni si suppone che, in prima approssimazione, gli autostati sono costanti. Si studia poi il problema con le due schedule sopra citate, quella lineare e quella trovata mediante la condizione di adiabaticità locale. In ambo i casi si è calcolato il tempo di annealing e la fidelity. Si nota che, come previsto dai calcoli, il tempo di annealig della schedule di Roland e Cerf è più basso dal caso lineare, e quindi minore di quello classico.

Grafichiamo le fidelity, partendo da quella dell'evoluzione lineare (fig. 2.13). Come si può notare dal grafico la probabilità di misurare lo stato fondamentale all'istante finale è pari a circa il 30%. Questo perchè nel punto di gap minima, ossia a metà evoluzione, la schedule è troppo ripida e permette in questo modo allo stato di eccitarsi, secondo Landau-Zener, e salire al livello energetico superiore.

La seconda immagine (fig. 2.14) invece è la fidelity dell'ultima schedule studiata. Questo grafico è completamente diverso dal precedente, infatti la probabilità associata a tale schedule è superiore al 99,99%. Anche qui vi è un minimo a metà dell'evoluzione che si aggira intorno al valore di 0,988, che è comunque trascurabile, e dipende dall'$\epsilon$ scelta.

Infine riportiamo un grafico che compara le due fidelity (fig. 2.12) sulla stessa scala.

Figura 2.21: Fidelity schedule lineare

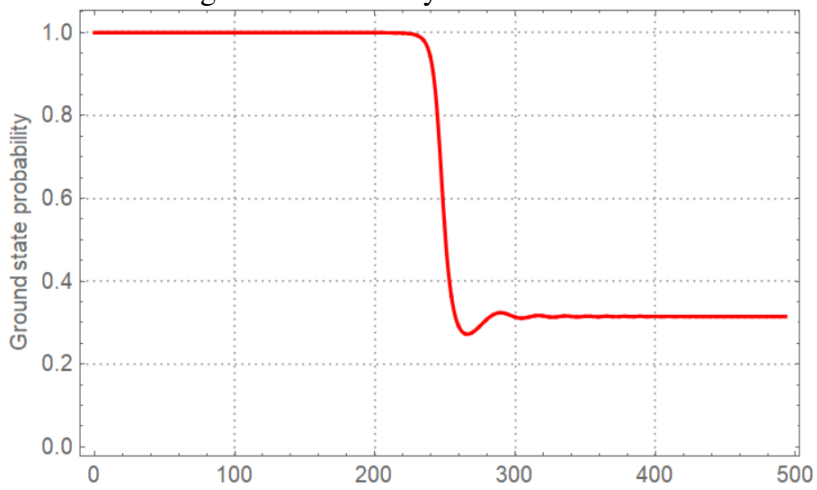



Figura 2.22: Fidelity schedule con adiabaticità locale

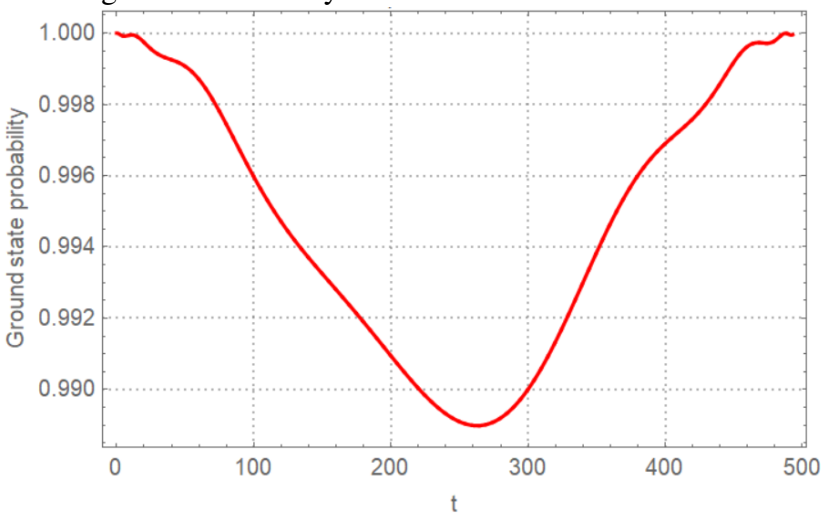
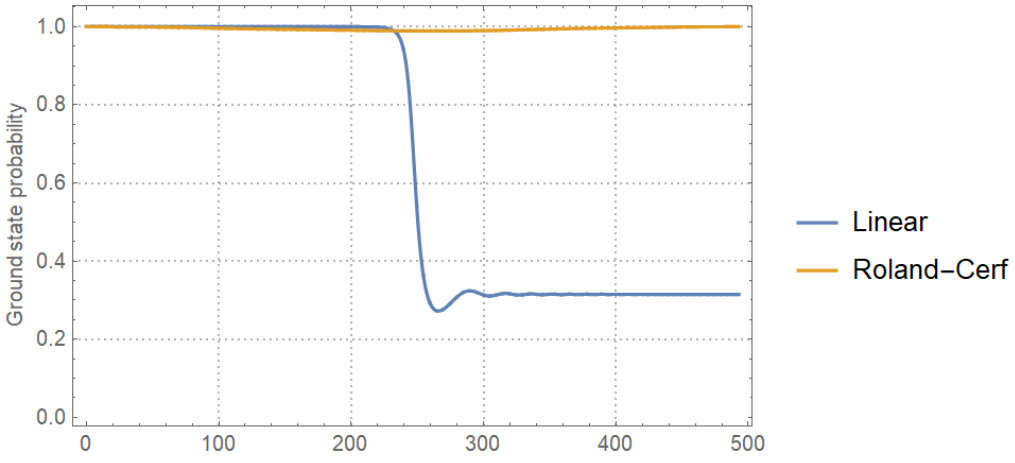


Figura 2.23: Comparazione delle fidelity



# Capitolo 3

# Conclusioni

Concludiamo ricapitolando i punti salienti di questo lavoro.
Inizialmente è stata data una breve descrizione della computazione quantistica, delle caratteristice principali di quest'ultima e delle fondamenta matematiche sulle quali si basa questa branca della fisica. Sono stati studiati 2 algoritmi di ricerca quantistica, l'algoritmo di Grover, basato su l'universal (o gate) quantum computation, e l'algoritmo di Roland e Cerf, basato sull'adiabatic quantum computation. Nel caso dell'algoritmo di Grover è stata descritta sia l'idea teorica che consente di trovare lo stato cercato mediante la tecnica dell' "amplificazione delle ampiezze", sia l'implementazione in termini di circuiti quantistici. Per l'algoritmo di Roland e Cerf, invece, è stato innanzitutto studiato il teorema sul quale si basa l'adiabatic quantum computation, ossia il teorema adiabatico. Utilizzando le nozioni derivanti da tale teorema è stato descritto l'algoritmo che sfrutta un hamiltoniana interagente per modificare uno stato iniziale affinchè, dopo un tempo di azione fissato, questo coincida con buona approssimazione con lo stato target. L'attenzione è stata puntata su una schedule grazie alla quale il tempo di annealing è minimizzato e la fidelity è massimizzata.
In entrambi i casi abbiamo effettuato anche delle simulazioni numeriche che sono in accordo con i calcoli teorici.
I risultati più importanti evidenziati sono l'equivalenza dei due algoritmi e la velocità di questi ultimi nel trovare la soluzione del problema di ricerca. Abbiamo dimostrato che la complessità computazionale in entrambi i casi è O($\sqrt{n}$), rispetto a O(N) del caso classico. Tutto ciò esprime l'enorme potenziale dei computer quantistici che in alcuni casi, come quello studiato, sono molto più efficaci di qualsiasi computer classico. Un problema rilevante, ad oggi, è trovare architetture in grado di costruire processori quantistici con un consistente numero di qbit. I record attuali si limitano a poche decine, nel caso della gate quantum computation, e a circa duemila nel caso dell'adiabatc quantum computation. Si tratta di numeri piccoli per applicazioni di largo interesse che, però, consentono di testare

su dispositivi fisici quantistici gli interessanti sviluppi teorici degli ultimi decenni.

# Appendice A

# Simulazione algoritmo di Grover

Di seguito è riportato integralmente l'algoritmo creato con Wolfram Mathematica per simulare l'algoritmo quantistico di Grover:

Condizioni iniziali

$$nqubit = 10$$
$$n = 2^{nqubit};$$
$$StatiBase = Permutations[Flatten[\{ConstantArray[1, \{nqubit\}],$$
$$ConstantArray[0, \{nqubit\}]\}], \{nqubit\}];$$
$$StatiBinariBase = Array[, n];$$

$$Do[$$
$$StatiBinariBase[[i]] = FromDigits[StatiBase[[i]], 2]$$
$$, \{i, 1, n\}]$$
$$StatiBinariBase;$$
$$Coefficients = ConstantArray[1, \{n\}];$$

$$Psi = \frac{1}{\sqrt{n}} Sum[Coefficients[[i]].StatiBase[[i]], \{i, 1, n\}];$$
$$II = IdentityMatrix[\{n, n\}];$$

Operatori della subroutine di Grover

$$ConstructU := Module[\{\},$$
$$U = ConstantArray[0, \{n\}];$$
$$Do[$$
$$U[[i]] = II;$$
$$U[[i]][[i, i]] = -U[[i]][[i, i]], \{i, 1, n\}];]$$
$$ConstructU$$

Target e numero di iterazioni

$U[[8]];$
$Elemento = 134;$
$steps = Floor[\frac{\pi}{4} Sqrt[n]]$
$psi0 = \frac{1.}{\sqrt{n}} Coefficients;$

$s = psi0;$
$w = ConstantArray[0, \{n\}];$
$w[[Elemento]] = 1;$
$sperp = s - w.sw;$
$sperpv = \{\{0, 0\}, \{1, 0\}\};$
$wv = \{\{0, 0\}, \{0, 1\}\};$
$sv = \{\{0, 0\}, \{s.sperp, s.w\}\};$
$svec[[1]] = sv;$
$svec[[2]] = sv;$

Subroutine di Grover

$Do[$
$psi1 = U[[Elemento]].psi0;$
$svec[[2(i) + 1]] = \{\{0, 0\}, \{psi1.sperp, psi1.w\}\};$
$psi0 = (2.Transpose[\frac{1.}{\sqrt{n}}\{Coefficients\}].\{\frac{1.}{\sqrt{n}} Coefficients\} - .II).psi1;$
$svec[[2(i) + 2]] = \{\{0, 0\}, \{psi0.sperp, psi0.w\}\};$
$v[[i + 1]] = psi0[[Elemento - 10; ; Elemento + 10]],$
$\{i, 1, steps\}]$

# Appendice B

# Simulazione algoritmo adiabatico di searching

Riportiamo in questa sezione un algoritmo implementato con Wolfram Mathematica per simulare l'algoritmo di searching adiabatico:

Funzione che ridà autovalori e autovettori ordinati in maniera crescente
$myEigensystem[mat\_] := Module[$
$\{e, v\},$
$\{e, v\} = Eigensystem[mat];$
$\{e, v\} = \{e[[\#]], v[[\#]]\}\&@Ordering[e];$
$\{e, v\}$
$]$

$Parameters$
$nspin = 3;$
$n = 2^{nspin}$

stato iniziale
$\psi 0 = ConstantArray[\frac{1.}{\sqrt{n}}, n]$
$Targetstate$
$\psi m = ConstantArray[0., n];$
$\psi m[[1]] = 1.;$

Hamiltoniana al tempo t=0
$H0 = IdentityMatrix[n] - KroneckerProduct[\psi 0, Conjugate[\psi 0]];$

Hamiltoniana al tempo t=T
$Hm = IdentityMatrix[n] - KroneckerProduct[\psi m, Conjugate[\psi m]]$

Hamiltoniana dipendente dal tempo
$H[s\_] := (1-s)H0 + sHm;$

Gap instantanea tra gli autovalori
$g[n\_, s\_] := \sqrt{1. - 4(1-\frac{1}{n})s(1-s)}$

Spettro;abbiamo stimato la rate di evoluzione dell'hamiltoniana per valutare la scala temporale per l'evoluzione adiabatica
$nn = 1001$
$\sigma = \frac{1}{nn-1.}; (*ds*)$
$spectrum = Array[\{\}, nn];$
$maxlist = Array[\{\}, nn];$
$dH = D[H[s], s];$
$Do[$
$ss = i\sigma];$
$\{e, v\} = myEigensystem[H[ss]];$
$spectrum[[i+1]] = Flatten[\{ss, e\}];$
$maxlist[[i+1]] = Max[Abs[Flatten[v.(dH/.s \to ss).Transpose[v]]]];$
$, \{i, 0, nn-1\}];$
$xx = spectrum[[All, 1]];$
$yy = Array[\{\}, n];$
$zz = Array[\{\}, n];$
$Do[$
$yy[[i]] = spectrum[[All, i+1]];$
$zz[[i]] = \{xx, yy[[i]]\};$
$zz[[i]] = Transpose[zz[[i]]];$
$, \{i, 1, n\}]$

Minima gap e rate di variazione di H[s];$\tau_{ad}$ è una stima del tempo adiabatico necessario per avere errore pari a $\epsilon$(e quindi una fidelity pari a 1-$\epsilon$) per una schedule lineare
$error = 0.1;$
$\delta min = (zz[[2, All, 2]] - zz[[1, All, 2]])//Min;$
$rate = Max@maxlist;$
$\tau AD = rate/(error \delta min^2);$
$TableForm[\{\{"\delta min", NumberForm[\delta min, \{6,6\}]\}, \{"(dH/ds)_{1,0}", NumberForm[rate, \{6,6\}]\},$
$\delta min = 0.353553$
$(dH/ds)_{1,0} = 0.935414$
$\tau_{AD} = 74.8331$

Il mimimo della gap vale $\delta_{min} = \sqrt{\frac{1}{n}}$

Schedule ottimale
$t[\epsilon\_, n\_, s\_] := \frac{1}{2\epsilon}\frac{n}{\sqrt{n-1}}(\arctan[\sqrt{n-1}(2s-1)] + \arctan[\sqrt{n-1}])$;
$s[\epsilon\_, n\_, t\_] := \frac{1}{2}\big[1 + \frac{\tan\frac{2\sqrt{n-1}t\epsilon - N\arctan\sqrt{n-1}}{n}}{\sqrt{n-1}}\big]$;
$T[\epsilon\_, n\_] := t[\epsilon, n, 1]$;

Possiamo restringerci a uno spazio $2x2$, generato da $\psi_m$ e il suo complemento ortogonale; In questo modo possiamo simulare un numero elevato di n senza determinare l'intero spazio di Hilbert
$Hreduced[n\_, s\_] := \{\{(1-s)(1.-\frac{1}{n}), -(1-s)\frac{\sqrt{n-1.}}{n}\}, \{-(1-s)\frac{\sqrt{n-1.}}{n}, s + \frac{1.-s}{n}\}\}$

$nspin = 10$;
$n = 2^{nspin}$;

Evoluzione in un intervallo infinitesimale
$unitaryStep[\psi\_, s\_, t\_, dt\_, \epsilon\_, n\_] := Module[$
$\{e, v, \psi out\},$
$\{e, v\} = myEigensystem[Hreduced[n, s[\epsilon, n, t + 0.5dt]]]$;
$\psi out = v.\psi$;
$\psi out = DiagonalMatrix[E^{-iedt}].\psi out$;
$\psi out = \psi out.v$;
$\psi out$
]

Successivamente compariamo la shedul lineare con quella ottimale.

Evoluzione temporale con la schedule lineare
Parametri relatici al tempo
$t0 = 0.$;
$tf = T[error, n]$;
$Nt = Floor[100 * tf]$;
$dt = \frac{tf-t0}{Nt}$;
$tlist = Table[t0 + idt, \{i, 0, Nt\}]$;
$slinear[\epsilon\_, n\_, t\_] := \frac{t}{tf}$;

Valori medi
$meanValues[\psi\_, ham\_] := Module[$

$\{e, v, tmpH, tmpfid, tmpres\}$,
$tmpH = ham$;
$\{e, v\} = myEigensystem[tmpH]$;
$tmpfid = Abs[v[[1]].\psi]]^2$;
$tmpres = Re[Conjugate[\psi]].tmpH.\psi] - e[[1]]]$;
$\{tmpfid, tmpres\}$
]

Condizioni iniziali
$\{e, v\} = myEigensystem[Hreduced[n, 0.]]$;
$\psi 0 = v[[1]]$;

$fidelity = Array[\{\}, Length@tlist]$;
$residual = Array[\{\}, Length@tlist]$;
$\psi = \psi 0$;
$\{fidelity[[1]], residual[[1]]\} = meanValues[\psi, Hreduced[n, 0.]]$;
$Do[$
$\psi = unitaryStep[\psi, slinear, tlist[[i]], dt, error, n]$;
$\{fidelity[[i+1]], residual[[i+1]]\} = meanValues[\psi, Hreduced[n, slinear[error, n, tlist[[i+1]]]]]$;
$, \{i, 1, (Length@tlist) - 1\}]$
Evoluzione con la schedule di Roland-Cerf
Parametri temporali
$t0 = 0.$;
$tf = T[error, n]$;
$Nt = Floor[100 * tf]$;
$dt = \frac{tf - t0}{Nt}$;
$tlist = Table[t0 + idt, \{i, 0, Nt\}]$;
Valori medi
$meanValues[\psi\_, ham\_] := Module[$
$\{e, v, tmpH, tmpfid, tmpres\}$,
$tmpH = ham$;
$\{e, v\} = myEigensystem[tmpH]$;
$tmpfid = Abs[v[[1]].\psi]^2$;
$tmpres = Re[Conjugate[\psi].tmpH.\psi - e[[1]]]$;
$\{tmpfid, tmpres\}$
]

Condizioni iniziali
$\{e, v\} = myEigensystem[Hreduced[n, 0.]]$;
$\psi 0 = v[[1]]$;

$fidelityRC = Array[\{\}, Length@tlist];$
$residualRC = Array[\{\}, Length@tlist];$
$\psi = \psi 0;$
$\{fidelityRC[[1]], residualRC[[1]]\} = meanValues[\psi, Hreduced[n, 0.]];$
$Do[$
$\psi = unitaryStep[\psi, s, tlist[[i]], dt, error, n];$
$\{fidelityRC[[i+1]], residualRC[[i+1]]\} = meanValues[\psi, Hreduced[n, s[error, n, tlist[[i+1]]]]];$
$, \{i, 1, (Length@tlist) - 1\}]$

# Bibliografia

[1] Michael A. Nielsen & Isaac L. Chuang *Quantum Computation and Quantum Information*

[2] Jérémie Roland and Nicolas J. Cerf *Quantum search by local adiabatic evolution*

[3] Messiah, A., 1962, *Quantum Mechanics, Vol. II (North-HollandPublishing Company, Amsterdam)*