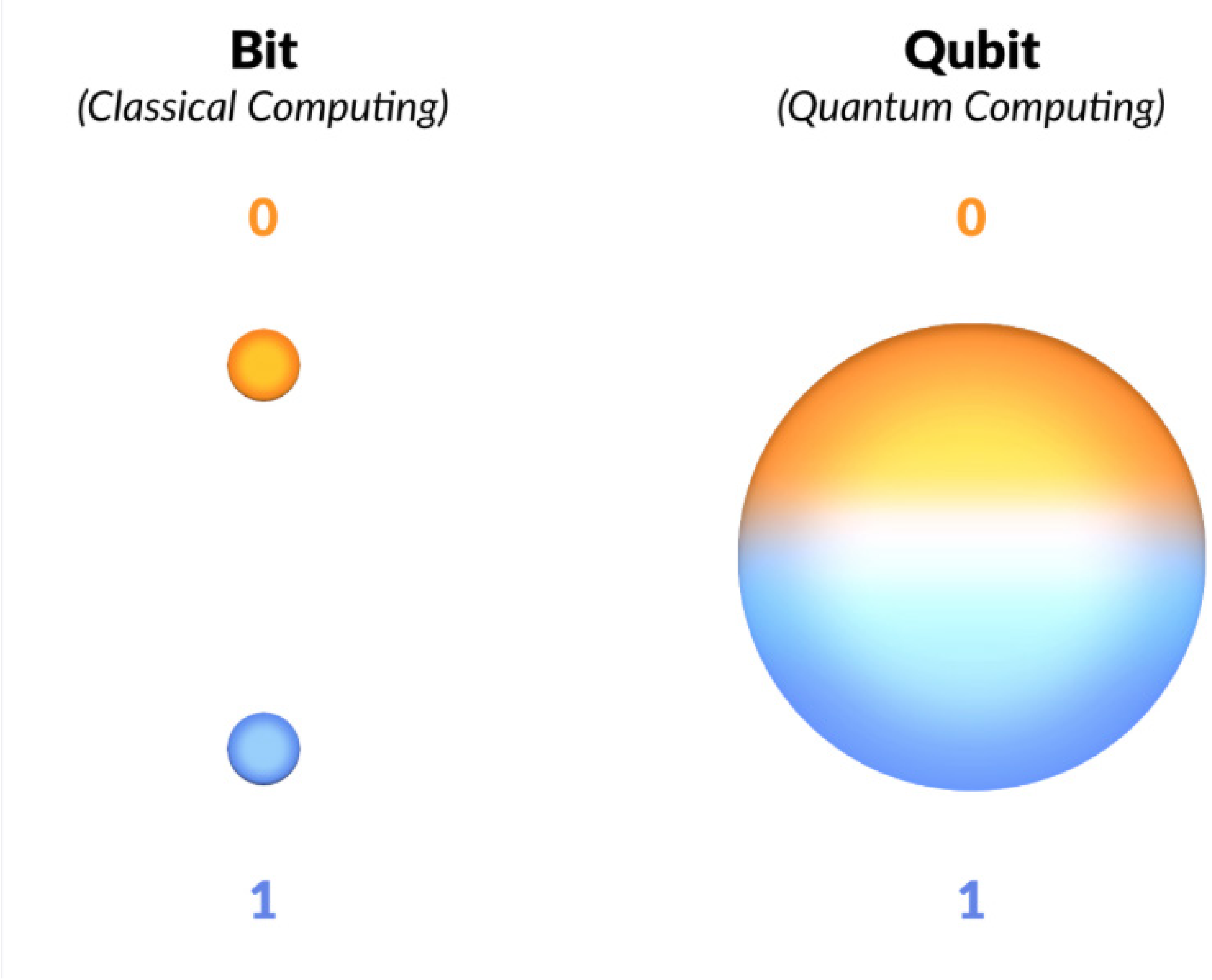
1 **Qubit** - Definizione

Un bit quantistico, o **qubit**, è l'equivalente nella computazione quantum di un bit classico. Come un bit è l'**unità minima di informazione** nella computazione classica, il qubit lo è nel mondo quantum.



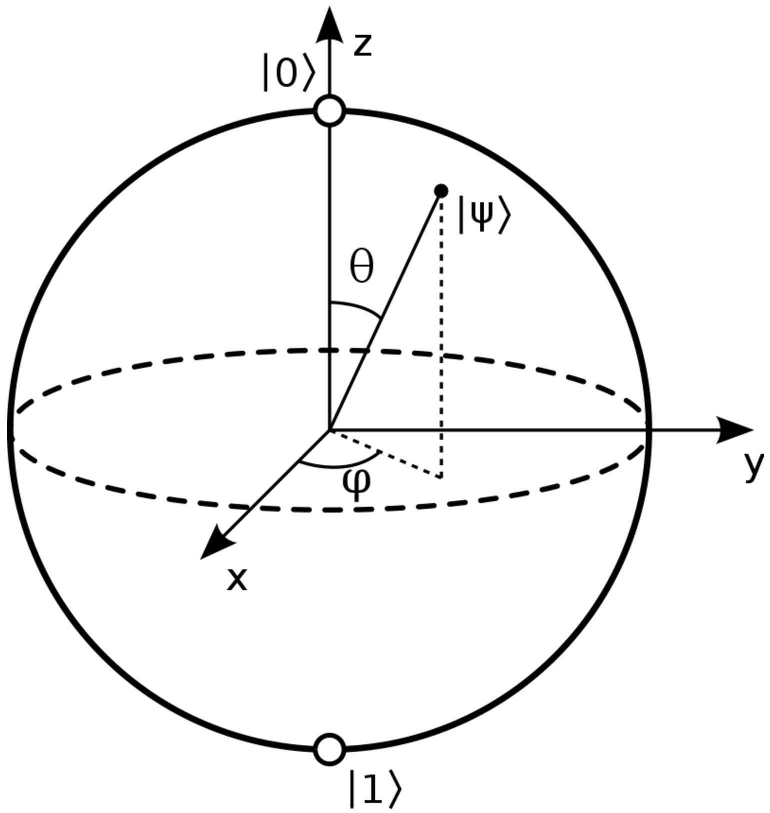
In un sistema classico, un bit può essere in uno stato solo. Al contrario, la meccanica quantistica permette ai qubit di essere in uno stato di **superposizione**. Questa proprietà dà alla computazione quantum un nuovo strumento per sbloccare una **nuova classe di algoritmi più efficienti**.

2 Qubit - Rappresentazione

Braket notation
Un qubit può essere interpretato come un **vettore** le cui componenti sono una **sovrapposizione della sua base ortonormale**. La base, nel nostro caso, è composta dai **possibili valori del qubit** dopo la misurazione, 0 o 1.

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,\|1\rangle \quad \alpha, \beta \in \mathbb{C}$$

Bloch sphere
Un qubit può essere interpretato come un **vettore unitario** in uno spazio 3D. Gli stati della base sono antipodali. Tutti gli **stati intermedi** sulla sfera unitaria sono una **sovrapposizione degli stati della base**, 0 e 1.



3 Quantum gate - Definizione

I quantum gate possono essere considerati come un **equivalente quantum delle porte logiche classiche**. Questi gate cambiando lo **stato del sistema** manipolando la sua **distribuzione di probabilità** piuttosto che applicare una **funzione logica**.

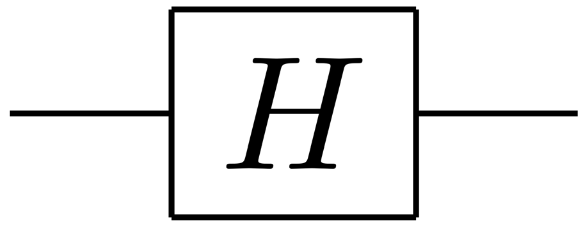
Questi gate devono essere **reversibili** in modo tale da preservare le proprietà della computazione quantistica. Quindi i gate saranno rappresentati da **matrici unitarie**.

$$U|a\rangle = \begin{bmatrix} u_{11} u_{12} \\ u_{21} u_{22} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$$

Queste matrici unitarie possono agire su **uno o più qubit** alla volta. Se agiscono su **n qubit** avranno una dimensione di $\mathbf{2^n \times 2^n}$.

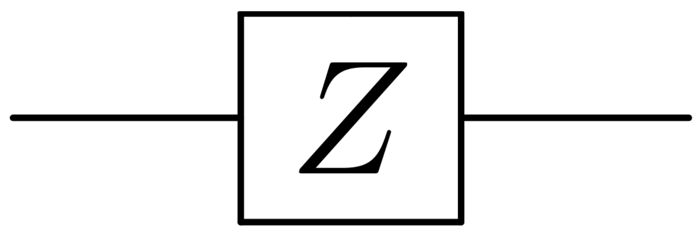
4 Quantum gate - singolo qubit

Hadamart gate
Pone il qubit in ingresso in uno stato di superposizione



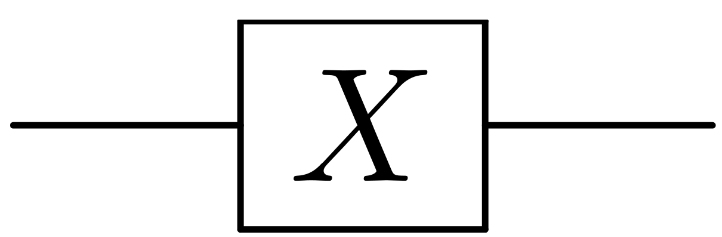
$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

Pauli-Z
Ruota la fase del qubit in ingresso di 180°



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

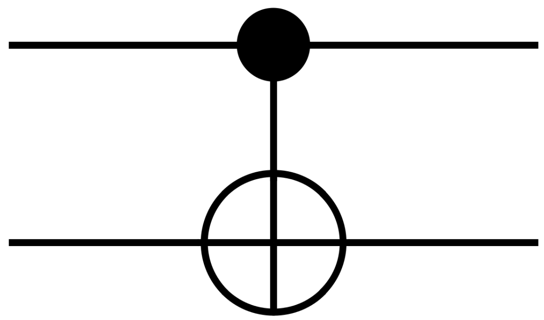
Pauli-X
Inverte lo stato del qubit in ingresso



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

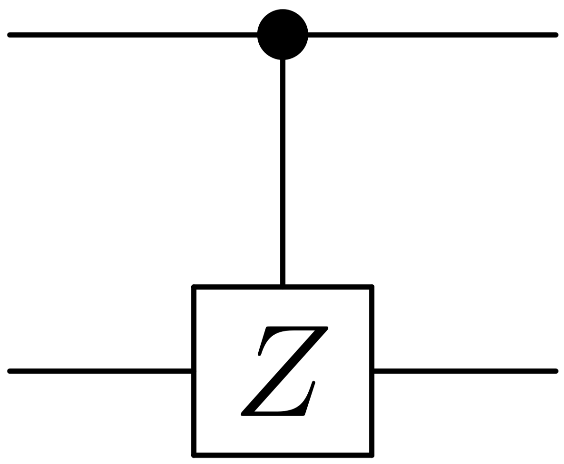
5 Quantum gate - Controllati

CNOT
Inverte il qubit in ingresso se il qubit di controllo è posto a 1



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

Controlled Z
Ruota la fase del qubit in ingresso di 180° se il qubit di controllo è posto a 1

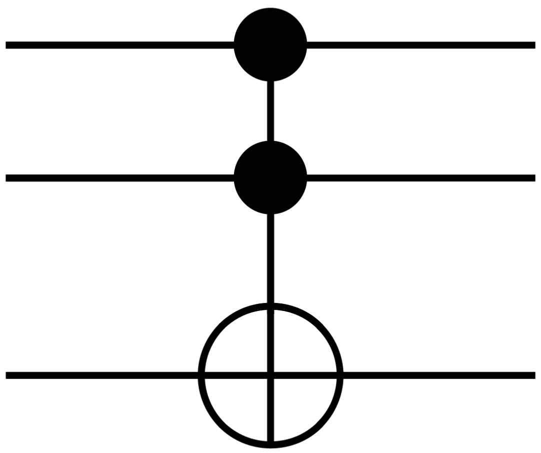


$$CZ = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

6 Quantum gate - Controllati a più qubit

I gate controllati possono essere **estesi ad un numero arbitrario di qubit** di controllo.

CCNOT/ Toffoli gate
Inverte il qubit in ingresso se I qubit di controllo sono posti a 1.



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

Se pur di estrema utilità i gate controllati a più bit sono di **difficile realizzazione pratica** in quanto richiedono un **numero esponenziale di gate** per la loro realizzazione.
Da notare che il **CCNOT è un gate universale** per la computazione classica, il che vuol dire che ogni **gate calssico** come AND, OR, XOR, ecc... può essere **ricostruito dal CCNOT**.
Questo prova che i computer quantistici sono almeno **Touring completi**.

7 Quantum speed up - Parallelismo

Il vantaggio dei computer quantistici è la loro abilità di **parallelizzare la computazione**. Consideriamo una funzione computabile con k gate classici.

$$f : \{0,1\}^n \rightarrow \{0,1\}^n$$

Questo circuito può essere implementato da **n CCNOT gate**. Chiameremo questo circuito U. Se **applichiamo l'operazione U** ad un ipotetico r**egistro posto in superposizione** avremo

$$|\psi\rangle = \frac{1}{\sqrt{2^n}}(|0\rangle + |1\rangle + |2\rangle + ... + |2^n - 1\rangle)$$

$$\Big\downarrow U$$

$$|\psi\rangle = \frac{1}{\sqrt{2^n}}(|f(0)\rangle + |f(1)\rangle + |f(2)\rangle + ... + |f(2^n - 1)\rangle)$$

Quindi la computazione della funzione viene eseguita un **numero di volte esponenziale** (possibili combinazioni dei qubit) facendo lo stesso numero di operazioni per **elaborare un singolo valore**.

1 Grover - Definizione

L'algoritmo di Grover è un **algoritmo quantistico** che trova con alta **probabilità** l'input ad una **black box** che produce un particolare output.

$$f : \{0, 1, ..., N-1\} \to \{0, 1\}, \quad where \quad N = 2^n,\ n \in \mathbb{N}$$

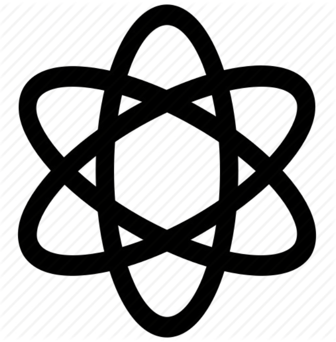
$$f(x) = \begin{cases} 0 & \text{if } x \neq w \\ 1 & \text{if } x = w \end{cases}$$

Per esempio si ipotizzi una funzione che mappa un insieme di interi all'insieme {0, 1}. La funzione sarà associata alla blackbox e Grover troverà quei input che avranno come risultato 1.

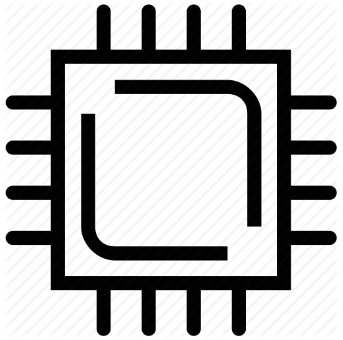
2

Grover - vantaggi (complessità)

Il più grande vantaggio nell'utilizzo di Grover è la usa **complessità** rispetto ad un algoritmo classico.



$$O(\sqrt{N}) * k$$



$$O(N)$$

Da notare che la **complessità** deigli **algoritmi quantistici** viene misurata in **chiamate all'oracolo e non temporalmente**. Inoltre bisogna considerare anche la complessità della **creazione dell'oracolo,** la quale dipende dal problema che si sta affrontando.

3 Grover - Fasi dell’algoritmo

Super position setup

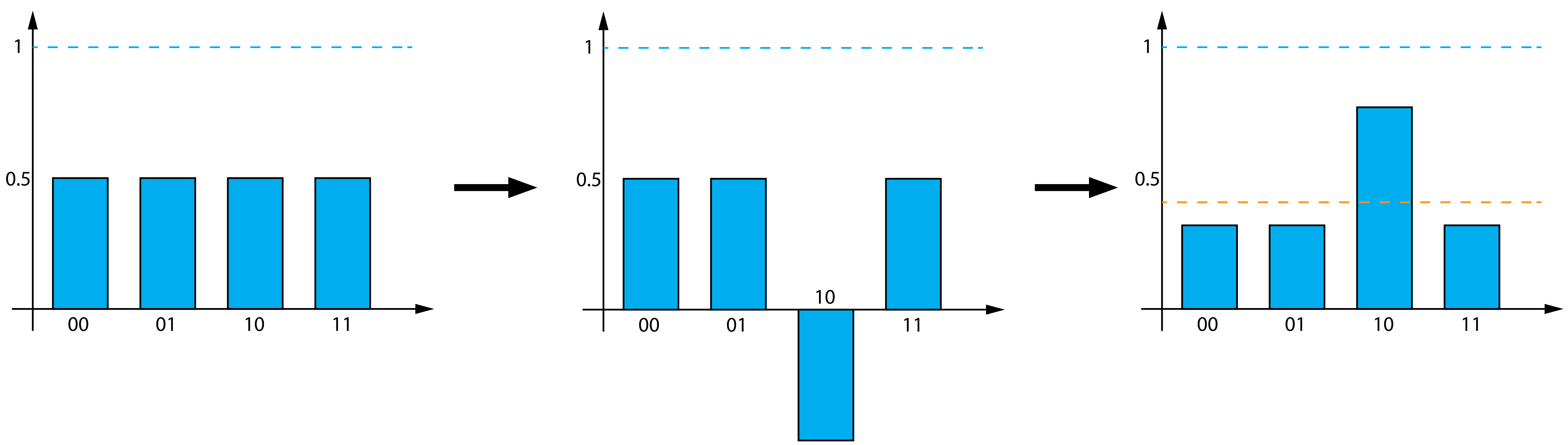
Il registro di input viene posto in **superposizione** attraverso i gate H. quindi ogni stato del registro sarà equiprobabile.

Phase invertion

Il registro di input viene elaborato dalla blackbox, detta anche **oracolo**, **invertendo la fase** dello stato ricercato dalla funzione codificata nell’oracolo.

Inversion about the mean

Tutti i possibili stati vengo **invertiti** rispetto all’**amplitudine media** degli stati. Così facendo si **aumenta la probabilità** che il registro si trovi nello stato ricercato



4 Grover - Iteration

Per ottenere un risultato migliore le ultime due fasi dell'algoritmo devono essere **iterate un numero preciso di volte.**

Numero iterazioni ottime singolo match

$$\frac{\pi}{4}\sqrt{N}$$

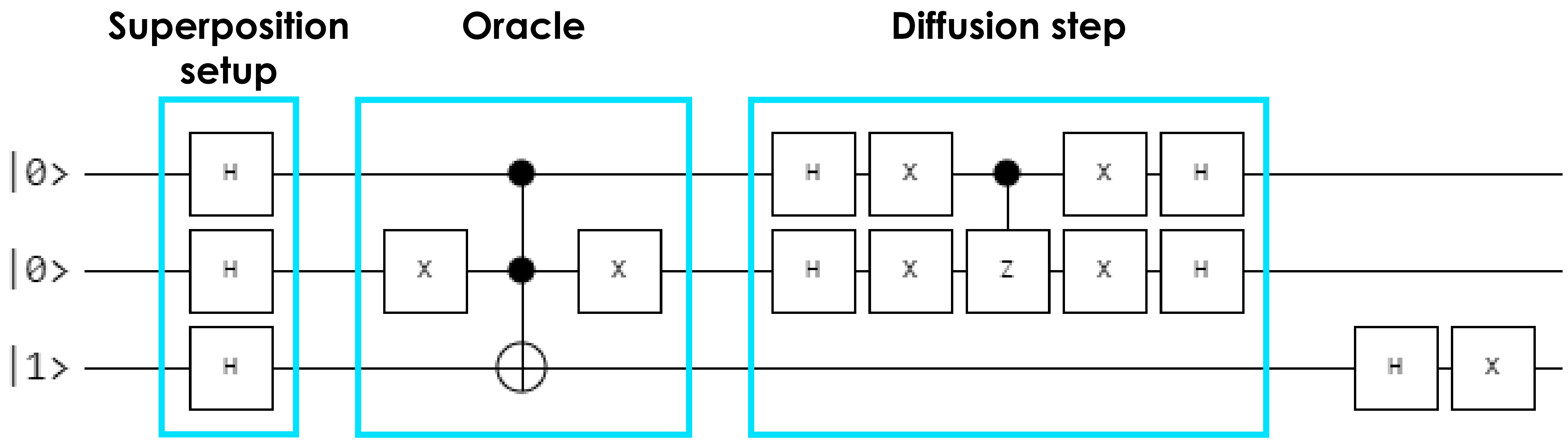
Numero iterazioni ottime m match

$$\frac{\pi}{4}\sqrt{\frac{N}{M}}$$

Il numero di iterazioni dipende da due fattori la **cardinalità** dei possibili stati del registro di input e il **numero di match**/stati ricercati. Spesso il numero di iterazioni ottime **non è un numero intero**. Questo porta ad una **minore probabilità di successo.**

5 Grover - implementazione

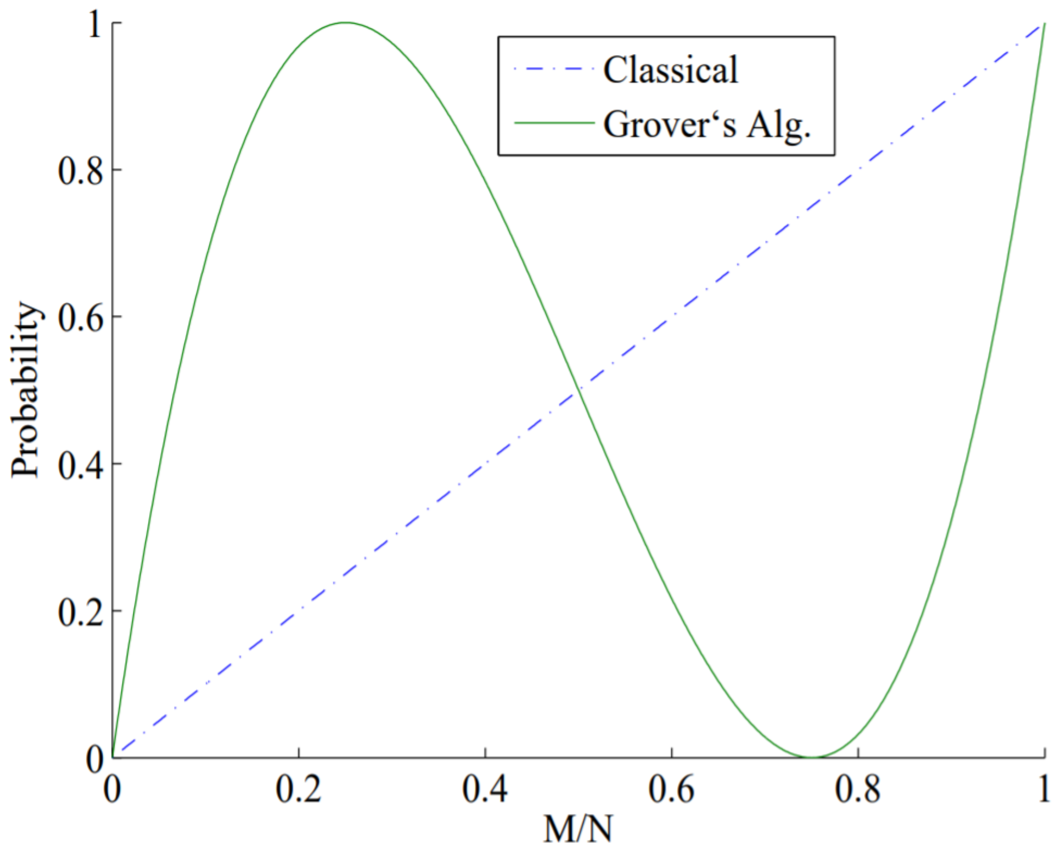
Prendendo in considerazione l'oracolo dell'esempio precedende, di seguito troviamo una possibile implementazione di Grover.



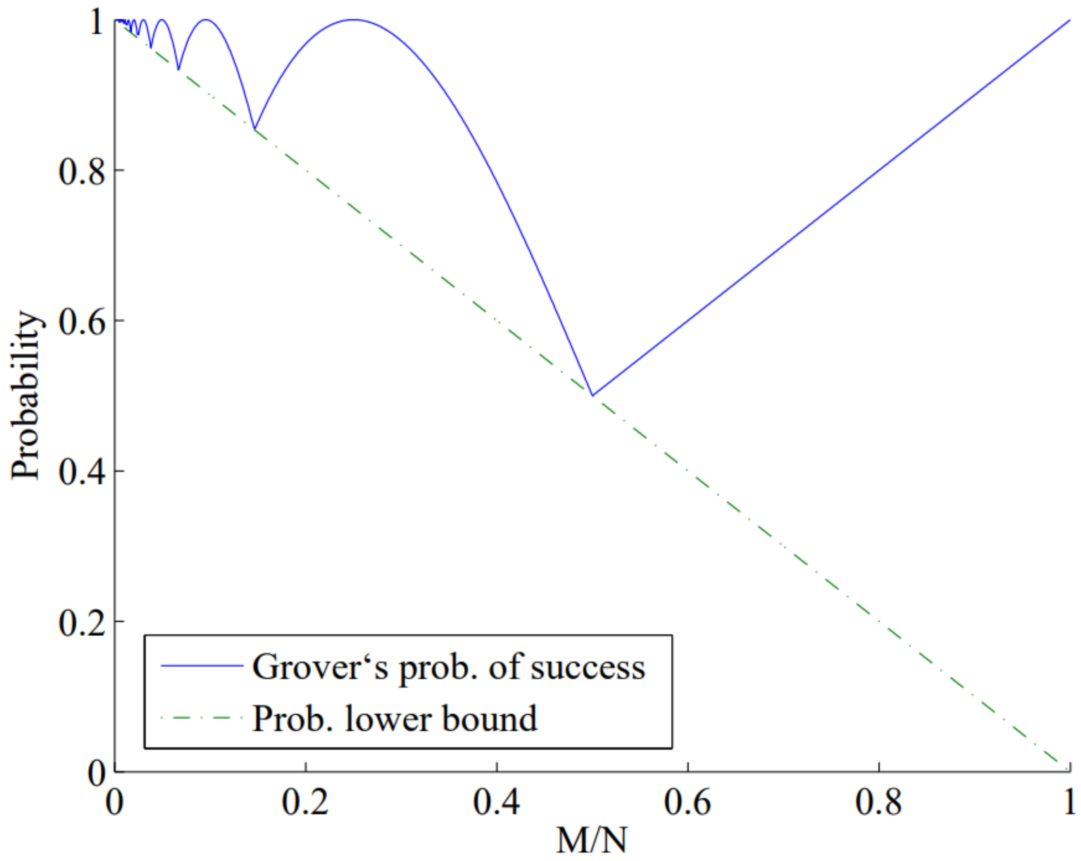
L'oracolo in questo caso codifica la funzione che ha come risultato 1 se l'input è 10.
Quindi il compito dell'**oracolo** è quello di **evidenziare gli stati ricercati** e il **diffusion step aumenta la probabilità** che il registro sia nello stato ricercato.

6 Grover - problematiche

Essendo un **algoritmo probabilistico** è bene aumentare il più possibile le possibilità di successo, ma non è sempre possibile. Infatti nel caso in cui **non si conoscono i numero di match**, l'algoritmo potrebbe essere iterato un numero errato di volte, portando ad un **risultato non attendibile** sistematicamente. Come si può vedere dai grafici sottostanti **all'aumentare dei match** Grover ha la **stessa probabilità di successo di un random guesser classico.**



Probabilità di successo di una singola iterazione
vs.
Random guesser classico



Probabilità di successo con iterazione ottime

7 Grover - MST

Grover può essere utilizzato nel calcolo di un MST per la **ricerca di archi** e la **valutazione dei loro pesi.** L'oracolo sarà una funzione che avrà come parametri l'**appartenenza** di un arco al grafo, la sua **direzione** e il suo **peso**