Come funziona un computer quantistico — la guida passo passo

Indice dei contenuti

- Come funziona un computer quantistico in sintesi?
- Cosa succede nella fase di preparazione dei qubit?
- Cosa sono le porte logiche quantistiche?
- Come si compone un circuito quantistico?
- Che ruolo hanno interferenza ed entanglement durante il calcolo?
- Come avviene la misura finale?
- Perché servono criogenia e isolamento?
- Con quali linguaggi e framework si programma?
- Come si verifica che il risultato sia corretto?
- Come si prova tutto questo senza hardware quantistico?
- Esegui il tuo primo circuito quantistico oggi

Il viaggio dell'informazione dentro un computer quantistico, dalla preparazione dei qubit fino alla misura finale, si può seguire anche senza una laurea in fisica, purché lo si affronti nell'ordine in cui accade. È l'ordine di questa pagina. Vedremo cosa succede in ciascuna delle tre fasi del calcolo, cosa sono davvero le porte logiche quantistiche e i circuiti che le compongono, che ruolo giocano sovrapposizione, interferenza ed entanglement, perché serve la criogenia e con quali linguaggi si programma. L'ultima sezione è dedicata a chi vuole mettere le mani su tutto questo pur non possedendo l'hardware, grazie all'emulazione quantistica.

Come funziona un computer quantistico in sintesi?

Le tre fasi sono preparazione, evoluzione e misura. Nella prima i **qubit** vengono inizializzati in uno stato noto, tipicamente $|0\rangle$. Nella seconda una sequenza programmata di porte logiche quantistiche, cioè il circuito, manipola ampiezze e fasi, così da creare le sovrapposizioni che servono a codificare il problema e ad orchestrare l'**interferenza** fra le diverse componenti dello stato. Nella terza lo stato viene interrogato e collassa in una stringa di bit classici, **che è l'unico output leggibile**. Poiché l'esito è probabilistico, il ciclo si ripete molte volte e la risposta emerge dalla distribuzione dei risultati. Tutto ciò che distingue un computer quantistico da uno classico accade nella seconda fase, che resta invisibile dall'esterno e, a differenza del calcolo classico, reversibile. Vale la pena tenere a mente questo schema mentre si leggono le sezioni seguenti, perché ogni algoritmo riempie quelle tre caselle in un modo diverso.

Cosa succede nella fase di preparazione dei qubit?

Ogni piattaforma ha un modo diverso di riportare i **qubit** a uno stato di partenza noto. Nei circuiti superconduttori si attende il rilassamento naturale verso lo stato fondamentale, oppure si applica un reset attivo che accorcia sensibilmente l'attesa. Negli ioni intrappolati il "pompaggio ottico" spinge gli elettroni nel livello desiderato, mentre nei fotoni la preparazione coincide con la generazione stessa. La qualità dell'inizializzazione viene misurata e dichiarata, perché una preparazione imperfetta inquinerebbe tutto il calcolo seguente. Da questo stato di quiete, il registro $|00\ldots0\rangle$, parte ogni circuito. Nelle specifiche di un computer quantistico commerciale la fedeltà di preparazione e di lettura compare accanto a quella delle porte, ed è uno dei parametri da leggere quando si confrontano macchine diverse.

Fedeltà tipiche di preparazione e lettura (SPAM) su hardware

> Per dare un'idea degli ordini di grandezza, sui processori superconduttivi l'errore di lettura si aggira sul 2% o 3%, mentre sui sistemi a ioni intrappolati scende sotto lo 0,1%, il che spiega perché le due famiglie di macchine vengano scelte per compiti diversi.

Cosa sono le porte logiche quantistiche?

Sono le istruzioni elementari della macchina, cioè trasformazioni unitarie, e quindi reversibili, dello stato dei **qubit**. Quelle di uso più comune formano un alfabeto ristretto. La **Hadamard** trasforma uno stato definito in una **sovrapposizione bilanciata**. Le rotazioni di fase, come le porte S e T e le rotazioni parametriche, ruotano le componenti dello stato e codificano informazione nelle fasi, che è il modo in cui l'algoritmo prepara il terreno all'**interferenza**. La CNOT, che agisce su due **qubit**, lega il bersaglio al controllo e genera entanglement. Un teorema garantisce che un insieme finito di porte, per esempio Hadamard, T e CNOT, sia universale, nel senso che qualunque calcolo quantistico si possa approssimare con precisione arbitraria usando soltanto quei tre mattoni. L'universalità è ciò che rende il computer quantistico una macchina programmabile in senso pieno, dato che lo stesso hardware esegue simulazioni di chimica o problemi di ottimizzazione cambiando soltanto la sequenza delle porte viste sopra.

Numero di qubit e fedeltà delle porte a uno e a due qubit oggi

> L'asimmetria fra i due tipi di porta è il dato che conta di più, perché le porte a un qubit sbagliano una volta su decine di migliaia, mentre quelle a due qubit, che sono poi quelle che generano entanglement, sbagliano da cento a mille volte più spesso, ed è su queste che si misura davvero la qualità di una macchina.

Come si compone un circuito quantistico?

Un circuito si rappresenta con una riga orizzontale per ogni **qubit** e con i simboli delle porte disposti in sequenza, mentre il tempo scorre da sinistra a destra. L'esempio minimo che genera correlazione quantistica richiede due sole istruzioni. Una **Hadamard** sul primo **qubit** e una **CNOT** tra il primo e il secondo producono lo

stato di Bell ($|00\rangle + |11\rangle)/\sqrt{2}$, una coppia perfettamente correlata. I circuiti reali aggiungono profondità, cioè strati di porte in successione, e poiché la **decoerenza** degrada lo stato mentre il calcolo procede, la profondità che si riesce a eseguire prima che il risultato diventi inutilizzabile è uno dei limiti pratici dell'hardware attuale, come discusso nel confronto fra computer quantistico e computer classico. Leggere un circuito si impara in fretta, mentre scriverne uno che sfrutti l'**interferenza** a proprio favore richiede molta più pratica.

Che ruolo hanno interferenza ed entanglement durante il calcolo?

L'**interferenza** è il meccanismo con cui l'algoritmo seleziona la risposta. Le componenti dello stato che portano a esiti sbagliati vengono messe in **controfase**, in modo che si cancellino a vicenda, mentre quelle corrette si sommano. L'**entanglement** fa qualcosa di diverso, perché tiene insieme il registro attraverso correlazioni fra **qubit** che nessuna descrizione indipendente dei singoli **qubit** riesce a catturare, e senza le quali un computer classico riuscirebbe a simulare il circuito in modo efficiente. Detto in termini pratici, un circuito privo di **entanglement** non sta calcolando nulla che un pc portatile non possa rifare: il numero di qubit comincia a pesare soltanto quando lo stato è realmente correlato, che è esattamente il caso in cui la simulazione classica diventa costosa.

Soglia oltre la quale la simulazione è impraticabile

> La barriera è di natura esponenziale, poiché descrivere esattamente n qubit richiede 2^n ampiezze complesse, cioè 16 GiB di memoria a 30 qubit, 16 TiB a 40 e 16 PiB a 50, ragione per cui la simulazione esatta dell'intero vettore di stato si ferma oggi intorno ai 40 qubit, salvo ricorrere a tecniche di compressione come le reti tensoriali. Questo limite vale per la simulazione a vettore di stato: le tecniche a reti tensoriali arrivano ben oltre questi valori.

Il formalismo del circuito e la regola di Born

> Lo stato di un registro di n **qubit** vive in uno **spazio di Hilbert** di dimensione 2^n e l'evoluzione complessiva prodotta dal circuito si scrive $|\psi_{fin}\rangle = U_m \cdots U_2 U_1 |00\ldots0\rangle$, dove ciascuna U_k è la matrice unitaria associata a una porta. Il disegno di quel prodotto di unitarie coincide con l'algoritmo. La misura nella base computazionale restituisce la stringa x con probabilità $p(x) = |\langle x|\psi_{fin}\rangle|^2$, cioè il modulo quadro dell'ampiezza associata a quell'esito, secondo la regola di Born. Vale la pena aggiungere una precisazione che il testo divulgativo di solito omette: l'**entanglement** è condizione necessaria ma non sufficiente per il vantaggio quantistico, perché il teorema di Gottesman-Knill mostra che esistono circuiti fortemente entangled, quelli composti da sole porte di Clifford, che una macchina classica riesce comunque a simulare in tempo polinomiale. Serve quindi **entanglement**, e serve che il circuito esca dal gruppo di Clifford, tipicamente grazie a **porte T**.

Come avviene la misura finale?

Interrogato nella base computazionale, ogni **qubit** restituisce 0 oppure 1 con probabilità pari al modulo quadro della relativa ampiezza, e la **sovrapposizione** svanisce. La lettura è essa stessa un processo fisico, con la sua fedeltà dichiarata, che si realizza raccogliendo fotoni da un risonatore nei circuiti superconduttori oppure

osservando la fluorescenza degli ioni nelle trappole. Il risultato di una singola esecuzione dice poco, e per questo il protocollo standard ripete il circuito centinaia o migliaia di volte, i cosiddetti **shot**, e ricostruisce la distribuzione degli esiti. Un buon algoritmo è quello che fa convergere quella distribuzione, con probabilità dominante, sulla soluzione ricercata. Il numero di shot è anche una leva economica concreta, perché sull'hardware commerciale si paga per esecuzione, e stimare in anticipo quanti shot servono per la precisione voluta è parte integrante del dimensionamento economico e temporale di ogni progetto.

Numero tipico di shot e costo per esecuzione

> Ad oggi (estate 2026) questi sono i dati commercialmente più eloquenti. Su Amazon Braket la tariffa per task è di 0,30 $ su tutte le macchine, mentre il prezzo per shot varia enormemente, da 0,00090 $ su Rigetti Ankaa e 0,00145 $ su IQM Garnet fino a 0,03 $ su IonQ Aria e 0,08 $ su IonQ Forte. Ne discende che un circuito da 100 shot costa 8,30 $ su IonQ Forte e 0,34 $ su Rigetti, mentre un'ottimizzazione variazionale da 200.000 shot arriva a 6.000 $ su IonQ Aria contro 85 $ su Rigetti. Va aggiunto che l'error mitigation su IonQ Aria impone un minimo di 2.500 shot per task.

Perché servono criogenia e isolamento?

Nelle macchine quantistiche fisiche la **decoerenza** è il fattore che limita pesantemente ogni calcolo. Qualunque scambio incontrollato di energia o di informazione con l'ambiente si comporta come una misura errata, non richiesta, e distrugge la **sovrapposizione** prima che il circuito sia terminato. I processori superconduttivi operano per questo intorno ai 15 millikelvin, una temperatura inferiore a quella della radiazione cosmica di fondo tra galassie, dentro criostati a diluizione che sono il componente più costoso e ingombrante dell'intera struttura hardware, mentre gli ioni isolati vengono confinati in camere a ultra-alto vuoto tramite trappole ioniche elettromagnetiche schermate da campi e vibrazioni. Non tutte le piattaforme richiedono il freddo estremo, dato che i fotoni si propagano a temperatura ambiente, anche se i rivelatori di singolo fotone più efficienti lavorano comunque in regime criogenico. I numeri aiutano a visualizzare le proporzioni, perché un criostato a diluizione pesa alcuni quintali e occupa una stanza, inoltre impiega decine di ore per raggiungere la temperatura operativa, mentre il chip che ospita misura solo pochi centimetri di lato. Per questa ragione la fase di studio e di prototipazione conviene svolgerla in altro ambiente, come vedremo nelle ultime sezioni.

Peso, ingombro, potenza frigorifera e consumo

> Anche qui i dati aiutano a visualizzare le proporzioni di questo hardware. Un criostato a diluizione di classe industriale smaltisce appena qualche decina di microwatt di calore alla temperatura di lavoro, e arriva al milliwatt soltanto a cento millikelvin, mentre richiede decine di litri di elio-3, una risorsa costosa e difficile da reperire. Il raffreddamento fino a dieci millikelvin impiega circa un giorno a sistema vuoto e sensibilmente di più quando il criostato è completamente cablato, e ogni installazione richiede un disegno di layout dedicato al locale che la ospita, mentre il chip che sta al centro di tutto questo misura pochi centimetri di lato.

Con quali linguaggi e framework si programma?

La programmazione avviene quasi sempre in linguaggio Python, attraverso framework open source che descrivono i circuiti a livello di porte. Qiskit, Cirq e PennyLane sono i più diffusi, affiancati da linguaggi dedicati come Q#. Il flusso di lavoro ricalca quello del software ordinario, perché si scrive il circuito, lo si esegue su un backend reale o emulato, si analizzano le distribuzioni di misura e si itera fino a ottenere il comportamento atteso. Gli standard intermedi come OpenQASM rendono i circuiti portabili tra piattaforme diverse. La sintassi si impara in pochi giorni, mentre ragionare per ampiezze e interferenze richiede parecchia pratica guidata. Una formazione strutturata su circuiti in emulazione accorcia di parecchio il tempo che serve a un team per prendere confidenza con questo modo di pensare.

Durata tipica di un percorso formativo aziendale

> L'esperienza formativa di HumanQ indica una forbice ampia, che dipende quasi per intero dal punto di partenza. Un team che padroneggia già i concetti di sovrapposizione, entanglement e interferenza, e che sviluppa abitualmente in Python, muove i primi passi in emulazione nel giro di poche settimane. Chi invece deve costruire da zero sia le basi concettuali sia la capacità di sviluppo ha bisogno di almeno sei mesi prima di lavorare in autonomia su circuiti propri.

Come si verifica che il risultato sia corretto?

Il carattere probabilistico dell'esito impone una disciplina di verifica che nel calcolo classico non serve. Per i problemi la cui soluzione è facile da controllare, come la fattorizzazione, si verifica classicamente la risposta proposta. Per gli altri si confrontano le distribuzioni statistiche ottenute con quelle attese su istanze ridotte, si usano circuiti di calibrazione e tecniche di mitigazione degli errori, e si stima la fedeltà complessiva dell'esecuzione. È qui che l'emulazione diventa uno strumento di verifica difficilmente sostituibile, perché la stessa sequenza di porte, eseguita in un ambiente esatto e privo di rumore, fornisce la distribuzione di riferimento contro cui giudicare l'hardware. Senza queste baseline, distinguere un errore di algoritmo da un errore di macchina o di disturbo sarebbe quasi impossibile. Vale anche in produzione, dove mantenere una pipeline di regressione in emulazione, eseguita a ogni modifica del circuito, è la pratica che distingue un progetto industriale effettivo da una mera dimostrazione.

Tassi di errore per porta e soglia richiesta

> L'ordine di grandezza attuale sulle porte a due qubit è di 10^{-3}, contro una soglia teorica del codice di superficie intorno all'1% e un valore operativo desiderabile ben al di sotto, poiché il costo in qubit fisici per qubit logico cresce rapidamente man mano che ci si avvicina alla soglia. Il confronto più efficace resta quello con gli algoritmi crittografici, dato che l'algoritmo di Shor richiede da milioni a miliardi di operazioni per fattorizzare gli interi alla base della crittografia RSA-2048, e con un errore dello 0,3% per porta gli errori si accumulano ben prima che il calcolo giunga a termine. Con un rapporto di 1.000 qubit fisici per qubit logico, una macchina da mille qubit implementa circa un solo qubit logico. Questo è il numero che spiega

meglio di ogni altro perché siamo ancora in era NISQ (Noisy Intermediate-Scale Quantum, ovvero calcolo quantistico su scala intermedia rumoroso).

Come si prova tutto questo senza hardware quantistico?

Con l'emulazione quantistica. La piattaforma di emulazione utilizzata da HumanQ riproduce l'intero ciclo descritto in questa pagina, dalla preparazione alle porte fino alla misura, su infrastruttura classica e ad alte prestazioni, inoltre offre due proprietà che l'hardware reale per sua natura non può ancora garantire. Lo stato è ispezionabile in ogni punto del circuito, ampiezze e fasi comprese, e ogni esecuzione è riproducibile alla perfezione. Si può seguire la **Hadamard** mentre crea la **sovrapposizione** e la CNOT mentre intreccia i **qubit**, verificando poi come l'**interferenza** selezioni l'esito. HumanQ sviluppa progetti di emulazione quantistica proprio per lavorare su ciò che è già realizzabile oggi, senza attendere la disponibilità di macchine fisiche, e per team di ricercatori e istituzioni questa è un'opportunità enorme per il calcolo quantistico operativo.

Dimensione massima dei circuiti emulabili

Il perimetro dimensionale si calcola in anticipo, poiché rappresentare esattamente lo stato di n qubit richiede 2^n ampiezze complesse. Una macchina stand-alone installata presso il committente lavora oggi su circuiti di 30 o 31 qubit, che occupano una trentina di gigabyte di memoria, ed è estendibile fino a 34 qubit con 256 gigabyte. Per un centro di ricerca o per un'amministrazione pubblica questa è la configurazione che conta davvero, perché l'intero ciclo di elaborazione, dati compresi, resta dentro il perimetro dell'ente e non attraversa mai un'infrastruttura di terzi. Chi non ha vincoli di questo tipo può salire ulteriormente in modalità SaaS, ma la scelta va fatta consapevolmente, dato che si scambia sovranità del dato con capacità di calcolo, magari inutilizzata.

Esegui il tuo primo circuito quantistico oggi

Contatta HumanQ e percorri tutte le fasi del calcolo quantistico, dalla preparazione alla misura, su casi d'uso reali e con il supporto del nostro team. Scopri l'emulazione quantistica, valuta la quantum readiness della tua azienda o contatta HumanQ.

Tag: entanglement, sovrapposizione, porte logiche, Hadamard, qubit, interferenza quantistica, misura quantistica, NISQ, emulazione quantistica.