Quantum Computing

Alessandra Di Pierro

Introduzione agli algoritmi quantistici

1 Computazioni classiche

Una differenza fondamentale tra i circuiti classici e quelli quantistici è che le porte logiche classiche potrebbero essere irreversibili (ad esempio `AND`, `XOR`, `NAND`), mentre le porte logiche quantistiche sono sempre unitarie e quindi reversibili. D'altra parte, sarebbe auspicabile che un modello di computazione alternativo sia in grado di esprimere almeno tutte le computazioni esprimibili con il modello classico. Il nostro primo obiettivo é quindi quello di rappresentare le computazioni classiche come trasformazioni unitarie, cioé come computazioni quantistiche. Poiché le trasformazioni unitarie sono invertibili (cioé reversibili), il primo passo da fare é quello di trasformare ogni computazione classica irreversibile in una reversibile. Per poter operare in modo reversibile è necessario che la funzione da valutare sia una biezione (i.e. iniettiva e surgettiva). In tal caso possiamo infatti risalire in maniera univoca da ogni output al valore dell'input che l'aveva generato, cioè operare in modo inverso. Una qualsiasi computazione irreversibile può essere trasformata in una equivalente computazione reversibile rendendo biunivoca la corrispondente funzione da valutare. Ad esempio, data una qualsiasi funzione:

$$f : \{0,1\}^k \mapsto \{0,1\}^m,$$

possiamo costruire $\tilde{f} : \{0,1\}^{k+m} \mapsto \{0,1\}^{k+m}$, tale che $\tilde{f}$ è biunivoca e calcola $(x, f(x))$ agendo sull'input $(x, 0^m)$, dove 0^m denota m bits inizializzati a 0. Ogni funzione biunivoca $f : \{0,1\}^n \mapsto \{0,1\}^n$, si puó in realtá vedere come una permutazione sugli n bits in input o, equivalentemente, sugli interi $0, 1, \ldots, 2^n - 1$. Di conseguenza, essa descrive una computazione classica reversibile.

Una qualsiasi computazione classica irreversibile si puó trasformare in una computazione equivalente ma reversibile usando la *porta di Toffoli*. Questa é un'operazione classica reversibile, rappresentata dal circuito in

IN			OUT		
a	b	c	a’	b’	c’
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

Table 1: Tabella di verità della porta di Toffoli

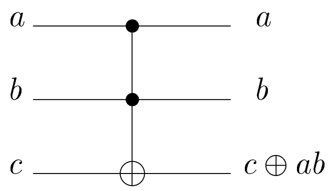

Figure 1: Rappresentazione della porta di Toffoli

Figura 1, che opera su tre bits in input: due sono bits di controllo e il terzo è il bit target che viene scambiato se i bits di controllo sono entrambi 1, come mostrato nella Tabella 1.

La reversibilitá di questa operazione si verifica facilmente osservando che applicando per due volte consecutive la porta di Toffoli si ottiene lo stesso risultato di partenza:

$$(a, b, c) \to (a, b, c \oplus ab) \to (a, b, c).$$

Quindi l’operazione stessa coincide con la sua inversa. Si verifica altrettanto facilmente che la porta di Toffoli rappresenta la permutazione $\pi = (67)$ sugli interi $0, 1, \ldots, 7$ (scambia la due sequenze 110 e 111).

La porta di Toffoli é universale per le computazioni classiche reversibili, cioé ogni computazione classica si puó costruire in modo reversibile mediante la porta di Toffoli. Questo risultato segue dall’universalità delle operazioni di `NAND` e `FANOUT` (l’operazione di copia di un bit classico) per le computazioni classiche e dal fatto che entrambe queste operazioni si possono esprimere mediante il circuito di Toffoli. Infatti, applicando l’operazione con $c = 1$, otteniamo $a' = a$, $b' = b$ e $c' = 1 \oplus ab = \neg ab$, cioè la simulazione di `NAND` come

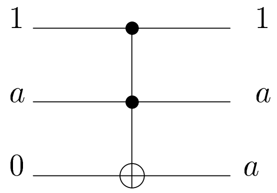

Figure 2: `FANOUT` realizzato mediante la porta di Toffoli

operazione reversibile. Il `FANOUT` reversibile si ottiene invece come mostrato in Figura 2: applicando la porta di Toffoli con $a = 1$ e $c = 0$ si ottiene come risultato la copia del bit b (Ricordiamo che questa operazione di copia non è possibile per un qubit).

Cosi come per `NAND` e `FANOUT` la costruzione di un circuito reversibile per una qualsiasi operazione classica f mediante la porta di Toffoli comporta l'utilizzo di alcuni bits di servizio in input (o *ancilla bits*) e in output (o *garbage*). Dopo aver eliminato questi bits di servizio, il circuito risultante esegue la trasformazione:

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

(dove x é l'input di f e y é il registro destinato a contenere l'output) e puó essere considerato come il *circuito reversibile standard* per la valutazione di f.

1.1 Computazioni classiche su circuiti quantistici

Come abbiamo giá osservato, una computazione classica reversibile corrisponde ad una permutazione sulle sequenze dei bit in input. Questo garantisce la possibilitá di costruire una matrice unitaria complessa che la rappresenta[1]. In particolare la porta di Toffoli può essere implementata come circuito quantistico. In questo caso l'input è dato da tre qubits e la trasformazione, analogamente al caso classico, consiste nello scambio del terzo qubit se i primi due sono 1. Ad esempio la porta di Toffoli quantistica applicata allo stato $|110\rangle$ produce lo stato $|111\rangle$. Un semplice esercizio é scrivere la matrice unitaria U corrispondente a questa permutazione.

[1]Un risultato della teoria dei gruppi e rappresentazioni assicura che esiste una rappresentazione, chiamata *rappresentazione unitaria standard*, del gruppo simmetrico delle permutazioni su 2^n simboli nel gruppo delle matrici unitarie complesse $2^n \times 2^n$. Tale rappresentazione associa ad una permutazione π la matrice U di elemento generico $U_{ij} = \delta_{i,\pi(j)}$, dove δ_{kl} denota il delta di Kronecker definito da $\delta_{kl} = 1$ se $k = l$ e $\delta_{kl} = 0$ altrimenti.

La porta di Toffoli quantistica si può quindi usare per simulare su un computer quantistico tutte le computazioni classiche, assicurando che un computer quantistico è in grado di eseguire una qualsiasi computazione eseguibile su un computer classico.

1.2 Computazioni probabilistiche su circuiti quantistici

Gli algoritmi *randomizzati* sono algoritmi che vengono eseguiti usando un generatore di numeri casuali (il lancio di una moneta) per decidere uno dei possibili rami di esecuzione. Il primo algoritmo randomizzato venne introdotto da Solovay e Strassen negli anni '70 per determinare se un numero é primo oppure no. L'algoritmo produce una risposta corretta solo con una certa probabilitá. Tale probabilitá si puó aumentare ripetendo l'esecuzione per un opportuno numero di volte.

Anche questi algoritmi possono essere simulati efficientemente mediante circuiti quantistici. Infatti, per simulare un bit random é sufficiente preparare un qubit nello stato $|0\rangle$ e applicare poi la porta di Hadamard. Si otterrá lo stato $(|0\rangle + |1\rangle)/\sqrt{2}$ che misurato dará 0 o 1 ciascuno con probabilitá $1/2$. Bisogna osservare inoltre che in questo modo si ottiene un numero "realmente" random, cosa che un computer classico non puó fare.

2 Parallelismo quantistico

Su un computer quantistico si può valutare una funzione $f(x)$ su valori differenti di x contemporaneamente. Questo è noto come *parallelismo quantistico* e rappresenta una caratteristica fondamentale dei circuiti quantistici.

Consideriamo una funzione booleana della forma:

$$f(x) : \{0,1\} \mapsto \{0,1\}.$$

Per calcolare $f(x)$ mediante una computazione quantistica si deve definire la trasformazione $f(x)$ come una trasformazione unitaria U_f. Come visto precedentemente, questo si puó fare applicando sullo stato di input $|x, y\rangle$, detto registro dei dati, un'appropriata sequenza di porte logiche quantistiche (che indicheremo con una scatola nera chiamata U_f) che trasformano $|x, y\rangle$ nello stato $|x, y \oplus f(x)\rangle$, detto registro target. Se $y = 0$ allora lo stato finale del secondo qubit conterrà esattamente il valore di $f(x)$.

Nel circuito in Figura 3, l'input è $\frac{|0\rangle+|1\rangle}{\sqrt{2}} \otimes |0\rangle$, cioè il valore di x è una sovrapposizione di 0 e 1 che si può ottenere applicando Hadamard a $|0\rangle$.

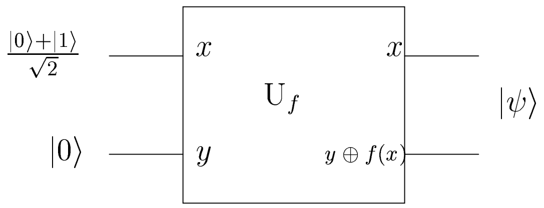

Figure 3: Circuito quantistico per valutare $f(0)$ e $f(1)$ simultaneamente

Applicando U_f a questo registro di dati si ottiene

$$\frac{|0, f(0)\rangle + |1, f(1)\rangle}{\sqrt{2}}.$$

Questo stato contiene informazioni sia sul valore $f(0)$ che sul valore $f(1)$. Abbiamo quindi valutato f simultaneamente su $x = 0$ e $x = 1$. Questo tipo di parallelismo è diverso da quello classico dove più circuiti (ognuno dei quali calcola $f(x)$ per un singolo valore di x) vengono eseguiti contemporaneamente.

Si può generalizzare questa procedura per calcolare funzioni su un numero arbitrario di bits usando una generalizzazione della porta di Hadamard nota come la *trasformata di Walsh-Hadamard*. Questa operazione consiste in n porte di Hadamard che agiscono in parallelo su n qubits. Per esempio, per $n = 2$, la trasformata di Walsh-Hadamard si indica con $\mathrm{H}^{\otimes 2} = H \otimes H$ e applicata a due qubits preparati nello stato $|0\rangle$ dà come risultato

$$\frac{|0\rangle + |1\rangle}{\sqrt{2}} \otimes \frac{|0\rangle + |1\rangle}{\sqrt{2}} = \frac{|00\rangle + |01\rangle + |10\rangle + |11\rangle}{2}.$$

In generale, il risultato di $\mathrm{H}^{\otimes n}$ applicato a n qubits nello stato $|0\rangle$ è:

$$\frac{1}{\sqrt{2^n}} \sum_x |x\rangle ,$$

dove x è la rappresentazione binaria dei numeri da 0 a $2^n - 1$ Quindi la trasformata di Walsh-Hadamard produce una sovrapposizione equiprobabile di tutti gli stati della base computazionale di n qubits. Osserviamo che per ottenere una sovrapposizione di 2^n stati servono soltanto n porte logiche H.

Esercizio 2.1 *Verifica che*

$$H^{\otimes n} |0\rangle^{\otimes n} = \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle .$$

Esercizio 2.2 *Verifica che*

$$\begin{aligned} H\left|x\right\rangle &= \frac{1}{\sqrt{2}}\sum_{y\in\{0,1\}}(-1)^{xy}\left|y\right\rangle \\ H^{\otimes n}\left|x\right\rangle &= \frac{1}{\sqrt{2^n}}\sum_{y\in\{0,1\}^n}(-1)^{x\cdot y}\left|y\right\rangle, \end{aligned}$$

dove $x \in \{0,1\}^n$ *e* $x \cdot y$ *è* $\sum_{j=1}^{n} x_i y_i$ *modulo* 2.

La valutazione parallela di una funzione, $f(x)$ con input x di n bits e 1 bit come output, può quindi essere eseguita da un circuito simile a quello in Figura 3, con $n+1$ qubits in input preparati nello stato $\left|0\right\rangle^{\otimes n}\left|0\right\rangle$. Si applica quindi Hadamard ai primi n qubits e successivamente il circuito U_f. Il risultato sarà

$$\frac{1}{\sqrt{2^n}}\sum_x \left|x\right\rangle\left|f(x)\right\rangle.$$

Il parallelismo quantistico non è direttamente utilizzabile nel senso che non è possibile ottenere tutti i valori calcolati con una sola misurazione: la misurazione dello stato $\frac{1}{\sqrt{2^n}}\sum_x \left|x\right\rangle\left|f(x)\right\rangle$ darà il valore di $f(x)$ per un *singolo* valore di x. Per sfruttare l'informazione nascosta in questo parallelismo dobbiamo in qualche modo sfruttare meglio l'informazione contenuta nella sovrapposizione $\frac{1}{\sqrt{2^n}}\sum_x \left|x\right\rangle\left|f(x)\right\rangle$, ad esempio utilizzando in maniera opportuna l'*interferenza* tra gli stati nella sovrapposizione. Combinando il parallelismo quantistico con questa proprietá che viene dalla meccanica quantistica si possono ottenere risultati come quello esemplificato dall'algoritmo di Deutsch.

2.1 Algoritmo di Deutsch

L'algoritmo di Deutsch mostra come attraverso la valutazione parallela di una funzione su tutti i suoi inputs si possano determinare proprietà globali della funzione come, per esempio, quella di essere una funzione *costante* o *bilanciata*.

Considera il circuito in Figura 4.

L'input del circuito che calcola la funzione f sono ora i qubits risultanti dall'applicazione di Hadamard agli stati $\left|0\right\rangle$ e $\left|1\right\rangle$. Tale input è quindi

$$\left|\psi_1\right\rangle = \frac{\left|0\right\rangle + \left|1\right\rangle}{\sqrt{2}} \otimes \frac{\left|0\right\rangle - \left|1\right\rangle}{\sqrt{2}}.$$

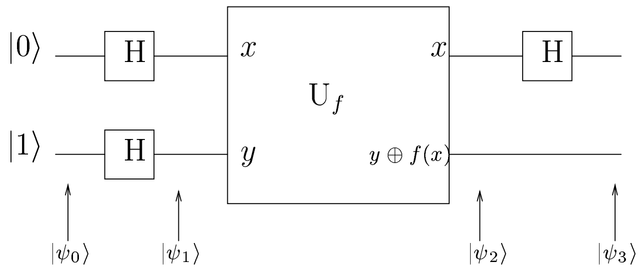

Figure 4: Circuito quantistico che implementa l'algoritmo di Deutsch

Applicando U_f allo stato $|x\rangle\,(|0\rangle - |1\rangle)/\sqrt{2}$ si ottiene $(-1)^{f(x)}\,|x\rangle\,(|0\rangle - |1\rangle)/\sqrt{2}$. Infatti, poichè U_f non modifica $|x\rangle$, se $f(x) = 0$ allora il risultato è $|x\rangle\,(|0\rangle - |1\rangle)/\sqrt{2}$, mentre se $f(x) = 1$ il risultato è $|x\rangle\,(|1\rangle - |0\rangle)/\sqrt{2}$.

Quindi, applicando U_f a $|\psi_1\rangle$ otteniamo un risultato $|\psi_2\rangle$ che dipende da due possibilità:

$$\begin{array}{ll} \text{se } f(0) = f(1) & \pm\left[\frac{|0\rangle+|1\rangle}{\sqrt{2}} \otimes \frac{|0\rangle-|1\rangle}{\sqrt{2}}\right] \\ \text{se } f(0) \neq f(1) & \pm\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}} \otimes \frac{|0\rangle-|1\rangle}{\sqrt{2}}\right]. \end{array}$$

Nota che nella seconda alternativa, si ha che $(-1)^{f(1)} = -(-1)^{f(0)}$. Nota inoltre che $|\psi_1\rangle$ si può scrivere come

$$\frac{1}{\sqrt{2}}\left(|0\rangle\,\frac{|0\rangle - |1\rangle}{\sqrt{2}} + |1\rangle\,\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right).$$

Pertanto U_f applicato $|\psi_1\rangle$ si può scrivere come

$$\frac{1}{\sqrt{2}}\left((-1)^{f(0)}\,|0\rangle\,\frac{|0\rangle - |1\rangle}{\sqrt{2}} + (-1)^{f(1)}\,|1\rangle\,\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right),$$

o, equivalentemente

$$\frac{1}{\sqrt{2}}\left((-1)^{f(0)}\,|0\rangle\,\frac{|0\rangle - |1\rangle}{\sqrt{2}} - (-1)^{f(0)}\,|1\rangle\,\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right),$$

cioè

$$\pm\frac{1}{\sqrt{2}}\left(|0\rangle\,\frac{|0\rangle - |1\rangle}{\sqrt{2}} - |1\rangle\,\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right),$$

ovvero

$$\pm\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}} \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}}\right].$$

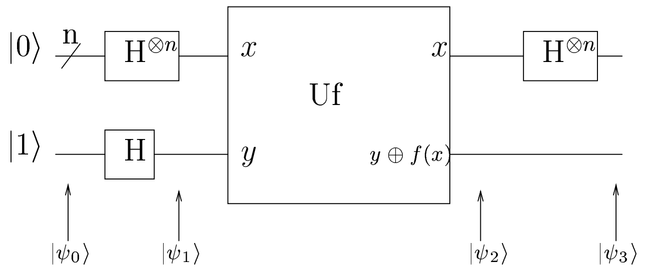

Figure 5: Circuito quantistico che implementa l'algoritmo di Deutsch-Jozsa

Applichiamo ora Hadamard al primo qubit e otteniamo $|\psi_3\rangle$ che risulta

$$\begin{array}{ll} \text{se } f(0) = f(1) & \pm |0\rangle \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] \\ \text{se } f(0) \neq f(1) & \pm |1\rangle \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]. \end{array}$$

A questo punto osserviamo che $f(0) \oplus f(1) = 0$ se $f(0) = f(1)$, altrimenti $f(0) \oplus f(1) = 1$. Possiamo quindi scrivere il risultato in maniera più coincisa come

$$|\psi_3\rangle = \pm |f(0) \oplus f(1)\rangle \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right].$$

Attraverso una misurazione del primo qubit possiamo quindi determinare con certezza (la probabilità associata al primo qubit è 1) il valore di $f(0) \oplus f(1)$ e quindi se la funzione f è *costante* oppure *bilanciata*. Per far questo abbiamo dovuto valutare f una volta sola.

2.2 Algoritmo di Deutsch-Jozsa

L'algoritmo precedente si può estendere a funzioni booleane su n bits. Consideriamo una funzione $f : \{0,1\}^n \mapsto \{0,1\}$ e supponiamo di sapere che f può essere o costante oppure *bilanciata* (cioè assume valore 0 su esattamente metà degli inputs e valore 1 sulla restante metà). Usando un algoritmo classico, nel caso peggiore abbiamo bisogno di valutare la funzione su almeno $2^{n-1} + 1$ valori per poter essere in grado di stabilire con certezza se f è costante o bilanciata. L'algoritmo quantistico di Deutsch-Jozsa ci permette di stabilirlo in un solo passo. Il circuito quantistico che implementa questo algoritmo è dato in Figura 5. A differenza del circuito in Figura 4, l'input x della funzione è dato da n qubits preparati nello stato $|0\rangle$, che chiameremo il registro dei dati. Il qubit target, destinato a contenere il risultato di $f(x)$, è invece preparato nello stato $|1\rangle$. Il registro iniziale è quindi

$$|\psi_0\rangle = |0\rangle^{\otimes n} |1\rangle \,,$$

dove $|0\rangle^{\otimes n}$ indica il prodotto tensore di n qubits $|0\rangle$.

Al registro dei dati $|0\rangle^{\otimes n}$ viene applicata la trasformazione di Walsh-Hadamard, $\mathrm{H}^{\otimes n}$, per produrre una sovrapposizione equiprobabile di tutti i 2^n stati della base computazionale.

Indicando con $x, y \in \{0,1\}^n$ sequenze di bits di lunghezza n, si dimostra che

$$H^{\otimes n} |x\rangle = \frac{1}{\sqrt{2^n}} \sum_{y \in \{0,1\}^n} (-1)^{x \cdot y} |y\rangle \,,$$

dove $x \cdot y$ è $\sum_{j=1}^{n} x_i y_i$ modulo 2 (cf. Esercizio4 dell'esercitazione 1). In particolare, risulta

$$H^{\otimes n} |0\rangle^{\otimes n} = \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle \,.$$

Gli stati $|\psi_1\rangle, |\psi_2\rangle, |\psi_3\rangle$ nel circuito in figura risultano quindi:

$$\begin{aligned}
|\psi_1\rangle &= \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] \\
|\psi_2\rangle &= \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} (-1)^{f(x)} |x\rangle \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] \\
|\psi_3\rangle &= \sum_{y \in \{0,1\}^n} \sum_{x \in \{0,1\}^n} \frac{(-1)^{x \cdot y + f(x)} |y\rangle}{2^n} \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right].
\end{aligned}$$

L'ampiezza di $|y\rangle = |0\rangle^{\otimes n}$ in $|\psi_3\rangle$ è $\sum_x (-1)^{f(x)}/2^n$. Consideriamo i due casi. Se f è costante, tale valore risulta 1 o -1. Questo significa (per la condizione di normalizzazione) che le ampiezze associate a tutti gli altri stati sono 0. Una qualsiasi misurazione del registro dei dati produce lo stato $|0\rangle^{\otimes n}$ con probabilità 1. Se invece f è bilanciata, l'ampiezza di $|y\rangle = |0\rangle^{\otimes n}$ risulta 0. Ciò significa che il risultato di una misurazione deve essere diverso da zero su almeno un qubit del registro dei dati. Dopo la misurazione siamo quindi in grado di stabilire con certezza se la funzione è costante o bilanciata.

Abbiamo ottenuto quindi una riduzione esponenziale della complessità rispetto ad un qualsiasi algoritmo classico.