Max-Flow

Quantum adiabatic computing

Nicola Barbaro (1070668) - Mario Bifulco (881727)

A.A. 2022/2023

Università degli studi di Torino - Ottimizzazione Combinatoria

Table of contents

Max-Flow

Problema di flusso massimo

Dato un grafo orientato $G = (V, E)$, anche chiamato *rete di flusso*, si richiede di trovare il valore massimo, del bene che si vuole schematizzare, in grado di fluire nella rete dal nodo sorgente s al nodo foce t.

Utilizzi tipici sono legati al trasporto di beni o l'instradamento su reti.

Formulazione matematica

$$
\begin{array}{llr}
\text{massimizza} & \displaystyle\sum_{(s,i)\in FS_i} x_{si} = x_{ts} & & (1)\\
\text{soggetto a} & \displaystyle\sum_{(h,i)\in BS_i} x_{hi} - \sum_{(i,h)\in FS_i} x_{ij} = 0 & \forall i \in V\backslash\{s,t\} & (2)\\
& \displaystyle(\sum_{(i,t)\in BS_t} x_{it}) - x_{ts} = 0 & & (3)\\
& \displaystyle x_{ts} - (\sum_{(s,i)\in FS_s} x_{si}) = 0 & & (4)\\
& 0 \leq x_{ij} \leq u_{ij} & \forall (i,j) \in E & (5).
\end{array}
$$

Vincoli del problema

1. Si vuole massimizzare il flusso su un arco *dummy*, senza capacità, che va dalla foce t alla fonte s;
2. Vincoli che permettono di rispettare la *conservazione dei flussi*;
3. Il flusso massimo trovato dal problema deve combaciare con la somma dei flussi entranti nella foce
4. Il flusso massimo trovato dal problema deve combaciare con la somma dei flussi uscenti dalla fonte
5. Bisogna rispettare il *vincolo di capacità*.

Problema del minimo taglio

Dato un grafo orientato $G = (V, E)$, si richiede di partizionare i vertici V in modo che:

1. Il nodo sorgente e quello foce non appartengano alla stessa partizione
2. Considerando N_s, la partizione contenente la sorgente, e N_t, la partizione del nodo foce, la somma degli archi con coda in N_s e testa in N_t deve essere minima.

Formulazione matematica

$$
\begin{aligned}
\text{minimizza} \quad & \sum_{(i,j)\in E} \omega_{ij} u_{ij} && && (1)\\
\text{soggetto a} \quad & \pi_t - \pi_s \geq 1 && && (2)\\
& \pi_i - \pi_j + \omega_{ij} \geq 0 && \forall (i,j) \in E && (3)\\
& \omega_{ij} \geq 0 && \forall (i,j) \in E && (4).
\end{aligned}
$$

Dove le variabili assumono valore:

$$
\pi_i = \begin{cases} 1 & i \in T \\ 0 & \text{altrimenti} \end{cases} \qquad \omega_{ij} = \begin{cases} 1 & (i,j) \in X_C \\ 0 & \text{altrimenti} \end{cases}
$$

Teorema Max-Flow Min-Cut

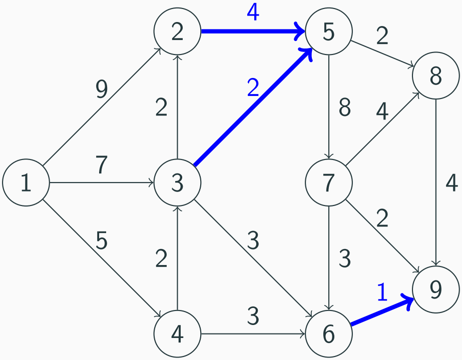
Teorema della dualità forte

Dato uno programma lineare primale P, se esso ammette soluzione ottimale x^*, allora anche il programma lineare duale D associato a P ammette soluzione ottima y^*, e in particolare si riscontra $y^* = x^*$.

Teorema Max-Flow Min-Cut

Il massimo valore di un flusso $s - t$ è uguale al taglio $s - t$ di capacità minima tra tutti i possibili tagli.

Esempio di un grafo di flusso



Dove il flusso massimo assume valore 7 e il taglio di capacità minima è composto dagli archi $\langle(2,5),(3,5),(6,9)\rangle$.

QAC

Computazione quantistica adiabatica

La computazione quantistica affronta i problemi in modo intrinsecamente diverso rispetto all'approccio classico.

La programmazione adiabatica ricerca la configurazione di variabili che minimizza l'energia del sistema fisico, ovvero una griglia di QuBit.

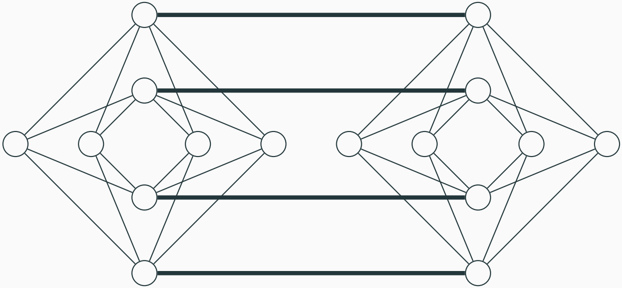
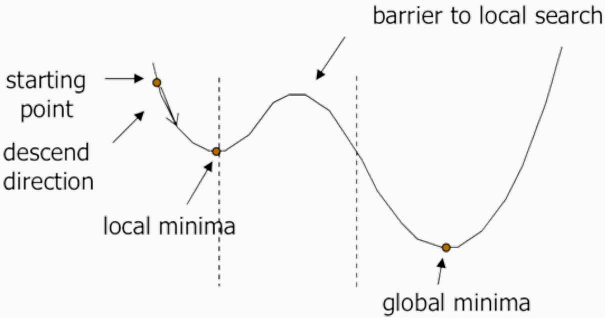


Figure 1: Esempio di QCPU a 16 qubit

Simulated Annealing

La ricerca effettuata tramite gli stati d'energia del sistema è approssimabile all'algoritmo di Simulated Annealing.



Problemi QUBO

Per essere eseguiti su macchine quantistiche i problemi devono essere riscritti come *problemi QUBO*.

Ovvero problemi composti da sole variabili binarie che assumono la forma:

$$\text{minimizzare} \underbrace{\begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix}}_{\bar{x}^T} \underbrace{\begin{bmatrix} a_1 & \cdots & a_n \\ \vdots & \ddots & b_n \\ 0 & \cdots & c_n \end{bmatrix}}_{Q} \underbrace{\begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}}_{\bar{x}}$$

Da CSP a QUBO

Min-Cut come problema QUBO

Per trasformare il problema Min-Cut in forma QUBO occorre rendere tutti i vincoli equazioni di somma zero, per cui i vincoli vengono riscritti come:

$$
\begin{aligned}
&\text{minimizza} && \sum_{(i,j)\in E} \omega_{ij} u_{ij} && && (1)\\
&\text{soggetto a} && \pi_t - \pi_s - s_1 - 1 = 0 && && (2)\\
& && \pi_i - \pi_j + \omega_{ij} - s_2 = 0 && \forall (i,j) \in E && (3)\\
& && \omega_{ij} - s_3 = 0 && \forall (i,j) \in E && (4).
\end{aligned}
$$

Conversione delle variabili

I problemi QUBO sono caratterizzati da variabili booleane, occorre quindi convertire le variabili di slack nella loro espansione binaria.

Il primo vincolo non permette a s_1 di assumere valori superiori a 1, è quindi sufficiente una variabile binaria y_1^0.

Per il secondo vincolo una cifra non è più sufficiente, occorrono due variabili, $s_2 = y_2^0 + 2y_2^1$.

Come per il primo vincolo, nel terzo caso possiamo sostituire s_3 con y_3^0.

Rilassamento Lagrangiano

I vincoli del problema sono trasformati in penalità sommate alla funzione obiettivo.

In questo modo si ottiene una singola equazione composta da variabili binarie e i rispettivi coefficienti.

Formulazione matematica

Dunque, riportiamo l'equazione del problema Min-Cut in forma QUBO:

$$\min z = \sum_{(i,j)\in E} \omega_{ij} u_{ij} + \lambda * ((\pi_t - \pi_s - y_1^0 - 1)^2 +$$

$$+ \sum_{(i,j)\in E} (\pi_i - \pi_j + \omega_{ij} - y_2^0 - 2y_2^1)^2 + \sum_{(i,j)\in E} (\omega_{ij} - y_3^0)^2)$$

Implementazione

Struttura del codice

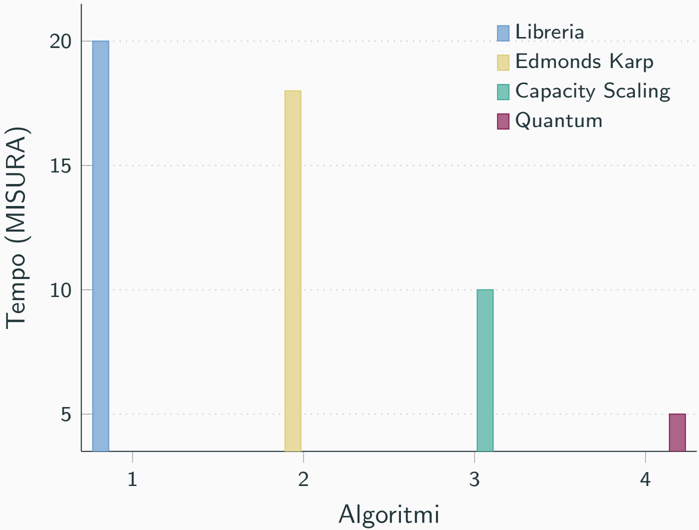
Sperimentazione svolta

L'implementazione Min-Cut in forma QUBO è stato eseguito sulla QCPU della D-Wave, confrontando i tempi d'esecuzione con:

1. Il metodo della libreria per il calcolo del flusso massimo
2. Una nostra implementazione dell'algoritmo di Edmonds-Karp
3. Una nostra implementazione dell'algoritmo di Capacity Scaling

Inoltre, è possibile svolgere un'indagine qualitativa per valutare pregi e difetti delle quattro strategie sperimentate.

Risultati ottenuti



Grazie per l'attenzione