GPU101 PiA Course
SpMV

Matteo Figini

2023

Introduzione

L'obiettivo del progetto è la velocizzazione del calcolo della moltiplicazione tra una matrice sparsa qualsisasi ricevuta in input e un vettore generato casualmente.
Sparse Matrix-Vector Multiplication (SpMV) e un kernel usato in diverse applicazioni scientifiche e ingegneristiche.

Basi Teoriche

Matrici Sparse e CSR

Le matrici sparse sono matrici che contengono un numero molto ristretto di valori non nulli in comparazione con quella che è la dimensione generale della matrice.
es.

$$\begin{matrix} 5 & 0 & 0 & 0 \\ 0 & 4 & 2 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 4 & 0 & 0 \end{matrix}$$

Per ottimizzare il salvataggio delle matrici sparse in memoria è stato utilizzato il CSR:
il formato CSR comprime la matrice in tre vettori:

- **Valori**: che contiene in ordine tutti i valori non nulli della matrice
  es.

$$[5, 4, 2, 3, 4]$$

- **Colonne**: che contiene gli indici colonna dei valori della matrice
  es.

$$[0, 1, 2, 2, 1]$$

- **Righe**: che contiene gli indici delle *'slices'* (fette) del vettore **Valori**. Ogni *'slice'* corrisponde a una riga.
  es.

$$[0, 1, 3, 4, 5]$$

Il Calcolo

$$\begin{matrix} 5 & 0 & 0 & 0 \\ 0 & 4 & 2 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 4 & 0 & 0 \end{matrix} \cdot \begin{matrix} a \\ b \\ c \\ d \end{matrix}$$

Il calcolo del prodotto Matrice-Vettore si compone di due fasi

- la moltiplicazione delle colonne della matrice per le righe del vettore
  es.

$$\begin{matrix} 5a & 0 & 0 & 0 \\ 0 & 4b & 2c & 0 \\ 0 & 0 & 3c & 0 \\ 0 & 4b & 0 & 0 \end{matrix}$$

- la somma dei valori nelle righe della matrice
  es.

$$\begin{matrix} 5a + 0 + 0 + 0 \\ 0 + 4b + 2c + 0 \\ 0 + 0 + 3c + 0 \\ 0 + 4b + 0 + 0 \end{matrix} = \begin{matrix} 5a \\ 4b + 2c \\ 3c \\ 4b \end{matrix}$$

Implementazione

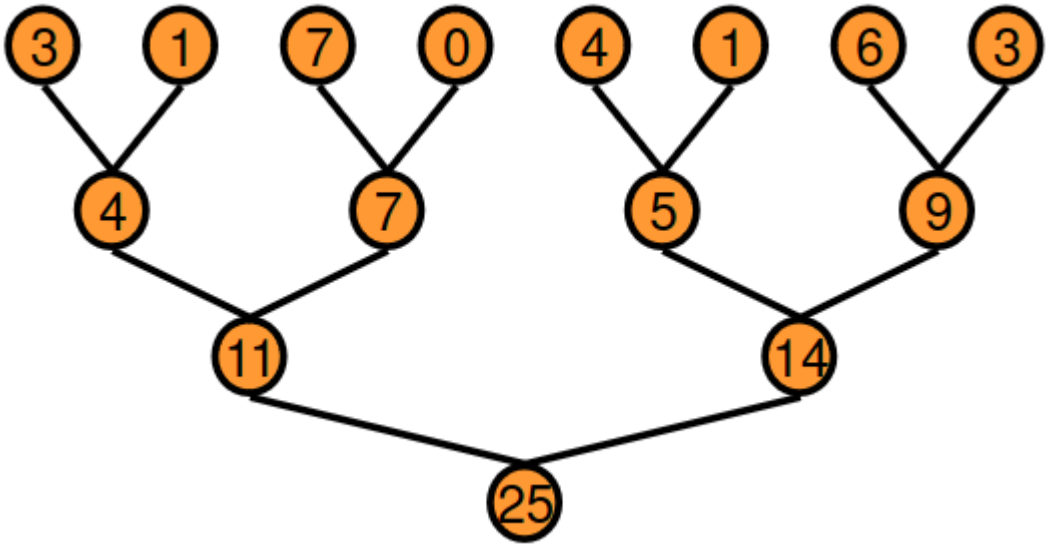
Seguendo il metodo di moltiplicazione ho diviso l'implementazione in due fasi

Moltiplicazione

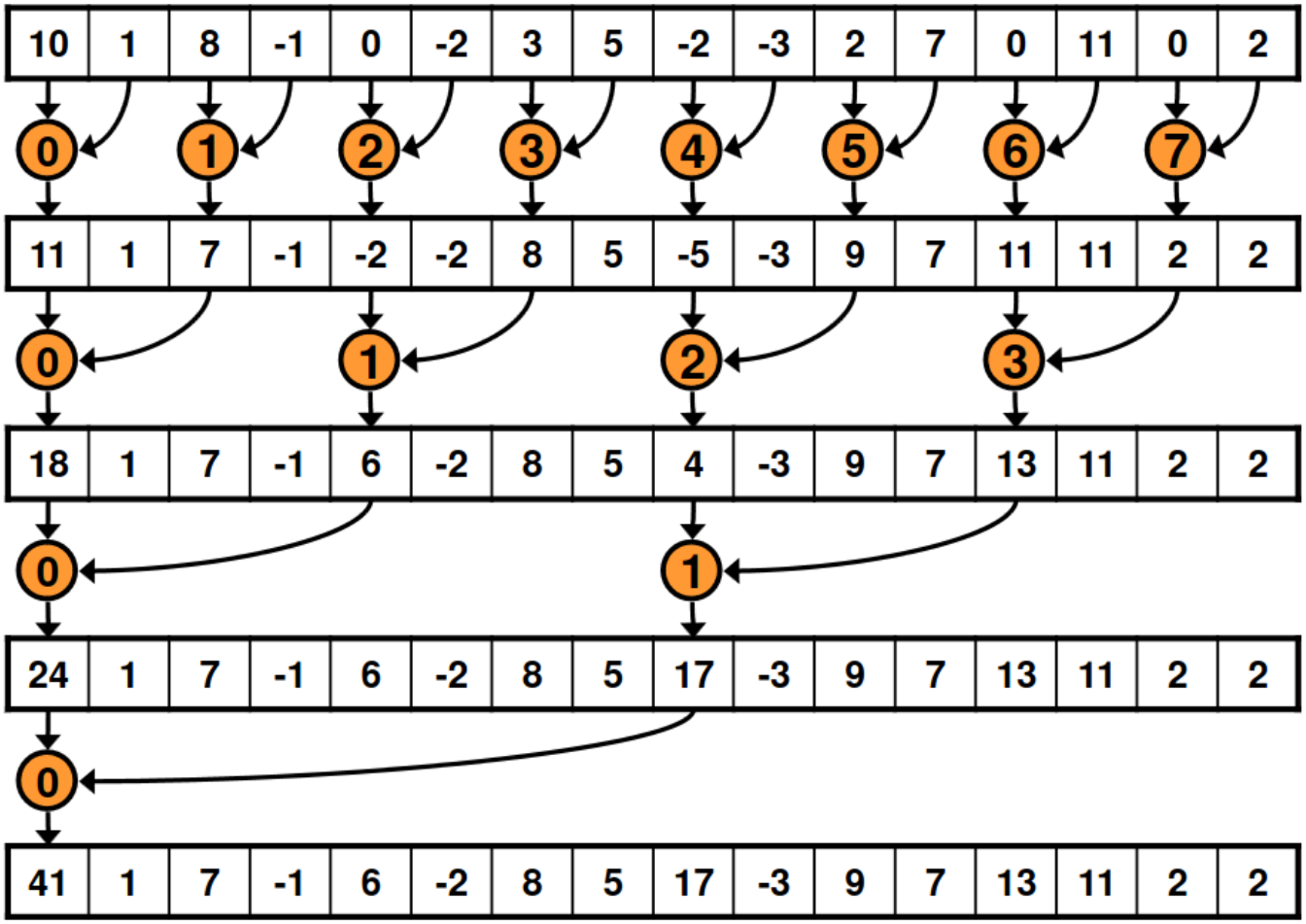
Ho assegnato ad
Per quanto riguarda la moltiplicazione quello che ho fatto è stato sfruttare la formattazione CSR.
Ho infatti assegnato ad ogni thread il valore presente nel vettore **Valori** corrispondende al suo indice. Sfruttando la corrispondenza tra i vettori **Valori** e **Colonne** della CSR si ricava immediatamente la riga del vettore da moltiplicare.

Somma - Parallel Reduction

La somma è una di quelle operazioni considerata *embarassing parallel*. Infatti ad ogni step le somme degli elementi sono indipendenti tra loro.
L'approccio utilizzato è un *tree based approach*. In cui accorpo l'operazione si somma a due a due all'interno dello stesso blocco.



In particolare ogni thread si occuperà di sommare due celle e salvare il risultato in quella più a sinistra.



Eseguendo la stessa operazione più volte si andrà ad accumulare il risultato della somma all'interno della primissima cella di memoria del vettore.