Sparse Matrix Transposition for GPUs

Massimiliano Incudini - VR433300
Michele Penzo - VR439232

***Sommario*—L'obiettivo principale di questo progetto è stato quello di implementare alcune metodologie proposte per effettuare *Sparse Matrix Transposition* su *Gpu*. Sono stati analizzati alcuni algoritmi, descritti in sezione II, partendo dall'algoritmo seriale, passando a cuSPARSE per finire con l'implementazione degli algoritmi descritti in [1]. Infine vengono esposti i risultati e tratte le conclusioni.**

I. Introduzione

II. Metodologie analizzate

In questa sezione vengono spiegate ed evidenziate le differenze tra le varie metodologie analizzate.

A. Trasposta seriale

La prima metodologia descritta è quella seriale. Sempre a partire dalla rappresentazione in formato *csr* della matrice iniziale l'algoritmo crea un array di elementi, dove per ogni colonna della matrice analizzata conta quanti elementi **nnz** ci sono. Possiamo definire questo array come un istogramma delle frequenze degli elementi delle colonne. Viene quindi applicato un algoritmo seriale di *prefix_sum* su questo array, che conterrà ora i valori corretti di **cscColPtr**. Infine gli indici di riga e i valori nel nuovo formato *csc* vengono sistemati.
Questa implementazione servirà come base sulla quale verranno eseguiti i controlli degli algoritmi successivamente implementati.

B. Nvidia cuSPARSE

Questo toolkit è implementato all'interno nelle librerie NVIDIA CUDA runtime. Le routine delle librerie vengono utilizzate per le operazioni tra vettori e matrici che sono rappresentate tramite diversi formati. Inoltre mette a disposione operazioni che permettono la conversione attraverso diverse rappresentazioni di matrici, ed inoltre la compressione in formato *csr* che è una delle più usate quando si vuole rappresentare matrici sparse in modo efficiente.
Il codice è stato sviluppato basandosi su due versioni di cuSPARSE a causa delle Gpu utilizzate. In fase di compilazione viene quindi controllata la versione usata: 9 o 10.
Nel caso in cui la versione usata sia la 10 vengono svolti alcuni ulteriori passi, viene effettuata l'allocazione dello spazio necessario e del buffer per il calcolo della trasposta. Per quanto riguarda la versione 9 invece questi passi non sono necessari. Infine viene chiamata la procedura che effettua il calcolo della trasposta.
Le procedure chiamate sono diverse in base alla funzione. Nel primo caso viene chiamata **cusparseScsr2csc**, mentre nel secondo caso **cusparseCsr2cscEx2**. Quest'ultima richiede come parametro anche l'algoritmo che viene utilizzato all'interno della procedura.
Dopo essere state eseguite entrambe ritornano i valori ottenuti tramite un'altro formato, *csc*, che ne esprime la rappresentazione tramite valori come csrColIdx, cscVal, cscColPtr, cscRowIdx. Infine viene controllata la correttezza e i tempi rispetto alle altre implementazioni.

C. ScanTrans

D. MergeTrans

III. Risultati

IV. Conclusioni

Riferimenti bibliografici

[1] K. H. W.-C. F. Hao Wang, Weifeng Liu, "Parallel transposition of sparse data structures," *ICS '16*, 2016.