Report
Sistemi di Calcolo Parallelo e Applicazione

Valerio Cristofori

AA 2021/2022

Abstract

In questo documento vengono argomentate le scelte effettuate per realizzare un nucleo di calcolo per il prodotto tra una matrice sparsa e un vettore.
Vengono poi presentati e analizzati i risultati del calcolo.

1 Introduzione

L'obiettivo è realizzare un nucleo di calcolo parallelo, con 2 tipi di parallelizzazione. La prima è basata su un'implementazione del concetto di multithreading su sistemi a memoria condivisa: OpenMP. La seconda è CUDA, ovvero un'API che permette l'uso di GPU per il calcolo parallelo (general-purpose computing su GPU). Ogni tipo di parallelizzazione viene poi confrontata con l'implementazione di un nucleo di calcolo seriale.
La fase di calcolo è preceduta da una fase di preprocessamento e memorizzazione dal formato `MatrixMarket` ai formati di memorizzazione per matrici sparse: `CSR` e `ELLPACK`.
Le matrici usate per il test sono alcune matrici della "Suite Sparse Matrix Collection", disponibili all'indirizzo `https://sparse.tamu.edu/`.

2 Preprocessamento

2.1 Caricamento e Formattazione

Sono state usate le funzioni di libreria fornite all'indirizzo `http://math.nist.gov/MatrixMarket/` per caricare le matrici dal formato `MatrixMarket` dove vengono specificati il numero di righe M, numero di colonne N, e il numero di non zeri NZ.
Da notare che per le matrici simmetriche viene memorizzato su file solo un triangolo (superiore/inferiore), di conseguenza, in memoria, vengono ricostruiti gli array di indici di

riga, colonna e valori dell'intera matrice, e aggiornato il numero dei *NZ*. È stato aggiunto un controllo sulle matrici memorizzate come *"pattern"*, popolando il vettore relativo ai valori nonzero di coefficienti di valore 1.0.

Sono stati formati in memoria i seguenti vettori:

I(1:NZ): Vettori degli indici di riga dei coefficienti nonzero

J(1:NZ): Vettori degli indici di colonna dei coefficienti nonzero

val(1:NZ): Vettori dei valori dei coefficienti nonzero

Viene poi costruito un vettore di indici **idxs** della matrice, in modo tale che l'i-esima posizione sia relativa a *I[i] * N + J[i]*. Vengono poi ordinati i vettori per indici di riga (si presuppone che i dati memorizzati su file siano ordinati per colonna). In questo modo il **idxs** risultante rappresenta gli indici dei valori nonzero della matrice, riga per riga.

2.2 Formattazione CSR

Compressed Storage by Rows. In questo caso la matrice viene formattata nel seguente modo:

IRP(1:M+1): Vettore dei puntatori all'inizio di ciascuna riga

JA(1:NZ): Vettore degli indici di colonna

AS(1:NZ): Vettore dei coefficienti

Viene costruito il vettore *IRP* partendo dal vettore *idxs*. È stato tenuto conto della possibile presenza di matrici con righe senza valori nonzero che portavano ad un calcolo errato (con valori di *IRP* a -1, come default): in questo caso le righe sono state scartate, ed è stato aggiornato il valore del numero di righe *M*.

2.3 Formattazione ELLPACK

In questo caso la matrice viene formattata nel seguente modo:

JA(1:M,1:maxnz): array 2D di indici di colonna

AS(1:M,1:maxnz): array 2D di coefficienti

La formattazione ELLPACK differisce dalla tipologia di parallelizzazione:

2.3.1 Formattazione ELLPACK per OpenMP

Nel caso di formattazione ELLPACK per parallelizzazione con OpenMP il valore di *maxnz* è fisso e pari al massimo valore di nonzeri per riga, su tutte le righe. Una volta trovato il valore *maxnz* si popolano i vettori *JA* e *AS*.

Da notare che per ogni riga che conta un numero di nonzeri minore di *maxnz* i rispettivi coefficienti sono nulli per *AS* e sono uguali all'ultimo indice valido incontrato lungo la

riga per JA.

2.3.2 Formattazione ELLPACK per CUDA

Per quanto riguarda CUDA invece oltre a trovare il valore $maxnz$ e inizializzare i vettori JA e AS viene costruito anche il vettore **MAXNZ(1:M)**, vettore di numeri di nonzeri per ogni riga.
Successivamente viene eseguita la trasposizione degli array 2D JA e AS. Rendendo trasposti i dati i coefficienti di ciascuna colonna risultano adiacenti in memoria essendo C un linguaggio con memorizzazione per righe.

3 Nucleo di calcolo

3.1 Calcolo Seriale

	Speedup CPU	
Matrice	CSR	ELLPACK
adder_dcop_32	0.21	0.01
af_1_k101	9.68	8.80
af23560	2.92	2.77
amazon0302	5.81	5.83
bcsstk17	1.71	1.07
cage4	0.00	0.00
cant	4.74	5.92
cavity10	0.53	0.47
cop20k_A	7.09	2.66
Cube_Coup_dt0	10.14	7.99
dc1	2.31	nd
FEM_3D_thermal1	2.28	2.39
lung2	3.26	2.43
mac_econ_fwd500	3.69	1.29
mcfe	0.23	0.18
mhd4800a	0.73	0.65
mhda416	0.12	0.04
ML_Laplace	9.96	9.01
nlpkkt80	9.74	8.75
olafu	3.39	3.20
olm1000	0.05	0.06
PR02R	8.47	4.58
raefsky2	1.48	1.23
rdist2	0.48	0.23
roadNet-PA	5.93	3.20
thermal1	2.31	3.44
thermal2	3.93	5.57
thermomech_TK	4.53	4.24
webbase-1M	6.09	nd
west2021	0.10	0.08

	Speedup GPU	
Matrice	CSR	ELLPACK
adder_dcop_32	1.78	0.78
af_1_k101	66.32	110.73
af23560	35.67	72.48
amazon0302	21.79	72.35
bcsstk17	26.97	46.71
cage4	0.00	0.00
cant	75.58	81.33
cavity10	6.81	17.19
cop20k_A	50.65	69.95
Cube_Coup_dt0	110.93	119.84
dc1	8.32	nd
FEM_3D_thermal1	28.77	61.26
lung2	9.68	24.02
mac_econ_fwd500	16.89	32.47
mcfe	2.41	4.21
mhd4800a	10.38	23.35
mhda416	1.25	2.27
ML_Laplace	113.11	115.12
nlpkkt80	69.91	109.88
olafu	53.04	79.07
olm1000	0.52	1.28
PR02R	81.70	65.85
raefsky2	18.29	39.91
rdist2	4.83	11.35
roadNet-PA	14.93	106.62
thermal1	16.81	39.09
thermal2	30.21	111.85
thermomech_TK	21.35	48.29
webbase-1M	13.11	nd
west2021	0.70	2.00

	GFlops CPU	
Matrice	CSR	ELLPACK
adder_dcop_32	0.04	0.00
af_1_k101	4.67	4.24
af23560	1.10	1.10
amazon0302	1.02	1.30
bcsstk17	0.85	0.49
cage4	0.00	0.00
cant	2.31	2.44
cavity10	0.21	0.17
cop20k_A	2.26	0.86
Cube_Coup_dt0	5.04	3.99
dc1	0.86	nd
FEM_3D_thermal1	0.92	0.94
lung2	1.07	0.49
mac_econ_fwd500	1.48	0.46
mcfe	0.08	0.04
mhd4800a	0.25	0.21
mhda416	0.03	0.03
ML_Laplace	4.83	4.36
nlpkkt80	4.67	4.24
olafu	1.65	1.42
olm1000	0.02	0.02
PR02R	4.14	2.25
raefsky2	0.46	0.39
rdist2	0.15	0.07
roadNet-PA	1.38	0.75
thermal1	1.17	1.02
thermal2	0.93	1.60
thermomech_TK	1.13	0.99
webbase-1M	1.66	nd
west2021	0.03	0.03

	GFlops GPU	
Matrice	CSR	ELLPACK
adder_dcop_32	0.44	0.19
af_1_k101	31.85	53.18
af23560	15.37	31.24
amazon0302	4.86	16.14
bcsstk17	12.07	20.91
cage4	0.01	0.01
cant	35.46	38.17
cavity10	2.88	7.27
cop20k_A	18.10	24.99
Cube_Coup_dt0	53.89	58.22
dc1	3.30	nd
FEM_3D_thermal1	13.05	27.79
lung2	3.75	9.29
mac_econ_fwd500	6.40	12.30
mcfe	1.00	1.74
mhd4800a	4.54	10.23
mhda416	0.43	0.78
ML_Laplace	53.51	54.45
nlpkkt80	33.36	52.43
olafu	24.76	36.91
olm1000	0.18	0.44
PR02R	38.61	31.12
raefsky2	8.17	17.83
rdist2	2.11	4.95
roadNet-PA	3.60	25.70
thermal1	5.55	12.91
thermal2	8.91	33.00
thermomech_TK	5.67	12.82
webbase-1M	3.89	nd
west2021	0.23	0.67

3.2 Calcolo con OpenMP

Before you type anything that actually appears in the paper, you need to include a `\documentclass{ouparticle}` command at the very beginning, and then the two commands that have to be part of any LaTeX document, `\begin{document}` at the start and `\end{document}` at the end of your paper.

3.3 Calcolo con CUDA

The main structure of your paper is as follows:

4 Performance

By default, all of the options within `article.cls` are available with this class file. This class file provides the following additional options.

4.1 Front matter

The title of the manuscript is simply specified by using the `\title{text}` command in the same manner as in this sample. Author's information consists of the name of the author and the corresponding institutions with addresses, as given in this example. Include an electronic mail address if available, inserting it into the `\email{text}` commands. You may follow the same coding if there are more than one author; separate authors with `\and`. Please identify the corresponding author with his/her electronic mail address by `\thanks{text}`. An abstract for your paper is specified by using `\abstract{text}`. A `\keywords{text}` macro may also be used to indicate keywords for the article. Use `\maketitle` after the abstract and keywords to make the header of your article.

4.2 Sections and subsections

To begin a new section, give the heading of that section in the `\section{text}` command. A section number is supplied automatically. Use the starred form (`\section*{text}`) of the command to suppress the automatic numbering. If you want to be able to make reference to that section, then you need to `label` it (see Section **??**). You can have sections up to five levels. The sectioning commands are `\section`, `\subsection`, `\subsubsection`, `\paragraph` and `\subparagraph`.

5 Conclusioni

The ends of words and sentences are marked by spaces. It does not matter how many spaces you type. The end of a line counts as a space. One or more blank lines denote the end of a paragraph.

5.1 Appendix

The `\appendix` command signals that all following sections are appendices, and therefore the headings after `\appendix` will be set as appendix headings. For a single appendix,

use `\appendix*` followed by the `\section{text}` command to suppress the appendix letter in the section heading.

6 References

The reference entries can be LaTeX typed bibliographies or generated through a BIBTeX database. BIBTeX is an adjunct to LaTeX that aids in the preparation of bibliographies. BIBTeX allows authors to build up a database or collection of bibliography entries that may be used for many manuscripts. They also save us the trouble of having to specify formatting. More details can be found in the *BIBTeX Guide*. For LaTeX reference entries use the `\begin{thebibliography}....\end{thebibliography}` environment (see below) to make references in your paper. We have provided the class file option to distinguish two styles of references. Those options are `numbib` and `nonumbib`. You can select one of these options with the `\documentclass` command. By default the class file will take the `numbib` option. The following is an example of LaTeX bibliography.

```
\begin{thebibliography}{0}
\bibitem{bib1}
Goossens, M., F. Mittelbach, and A. Samarin: {\em The {\LaTeX} Companion}.
Addison-Wesley, Reading, MA, USA, 1994.
\bibitem{bib2}
Knuth, D.E: {\em The {\TeX}book}. Addison-Wesley, Reading, MA, USA, 1984.
\bibitem{bib3}
Lamport, L.: {\em {\LaTeX} -- A Document Preparation System -- User's
Guide and Reference Manual}. Addison-Wesley, Reading, MA, USA, 1985.
\bibitem{bib4}
Smith, I.N., R.S. Johnes, and W.P. Hines: 1992, 'Title of the Article',
\textit{Journal Title in Italics} \textbf{Vol. no. X}, pp. 00--00
\end{thebibliography}
```

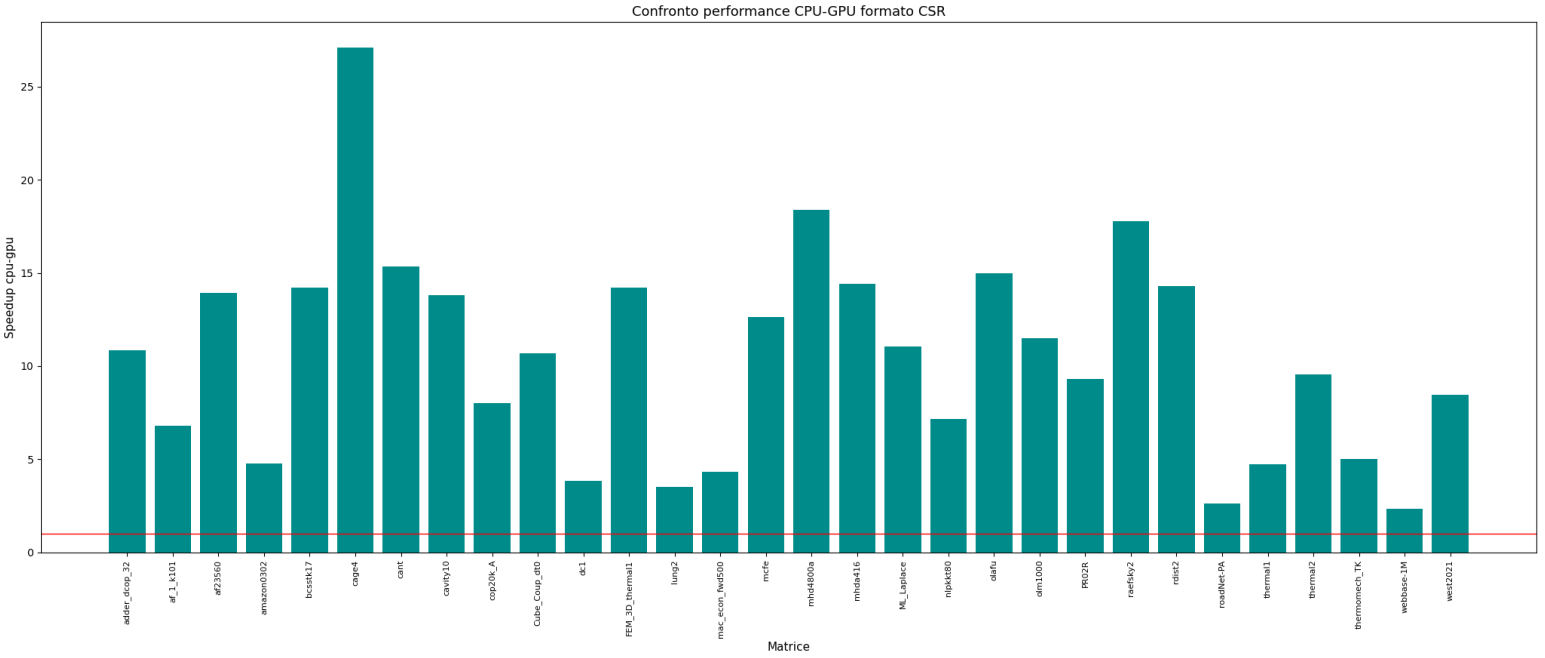

Figure 1: Speedup tra CPU e GPU del calcolo con formato CSR

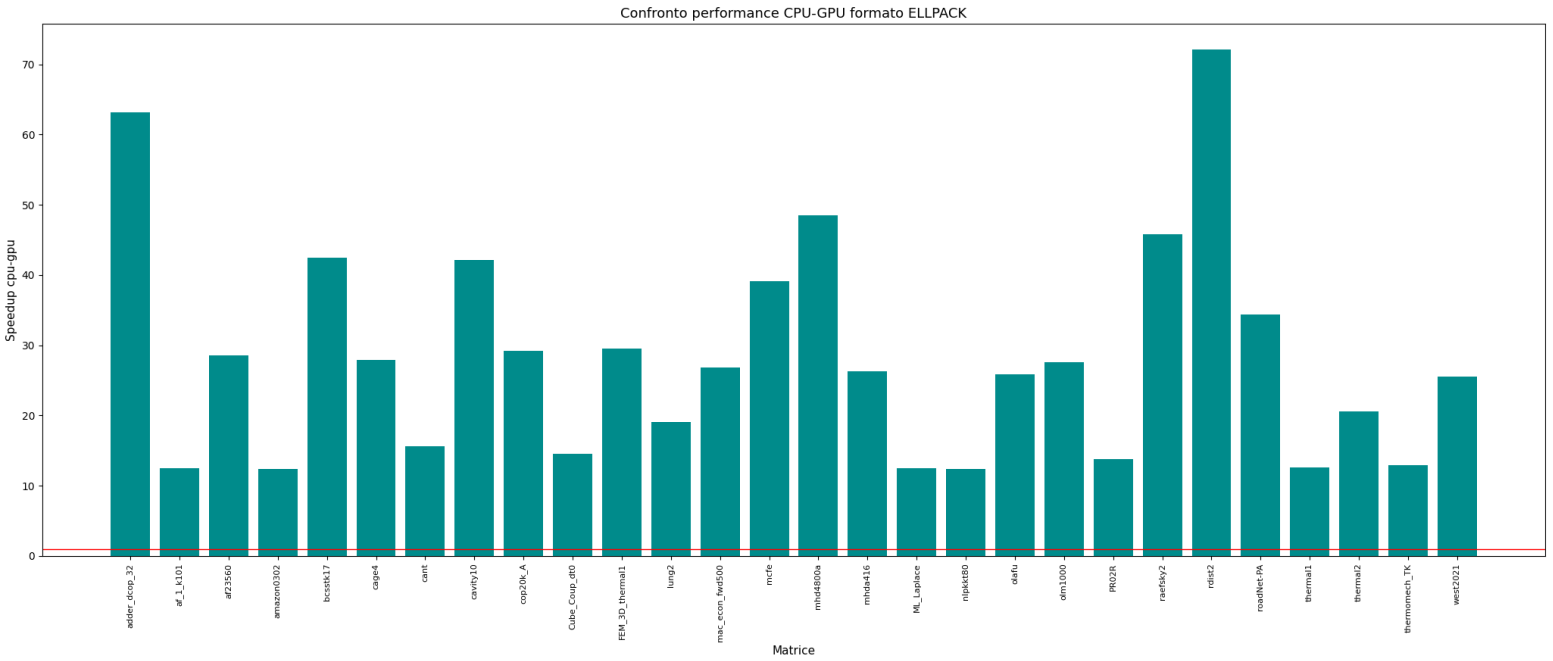

Figure 2: Speedup tra CPU e GPU del calcolo con formato ELLPACK

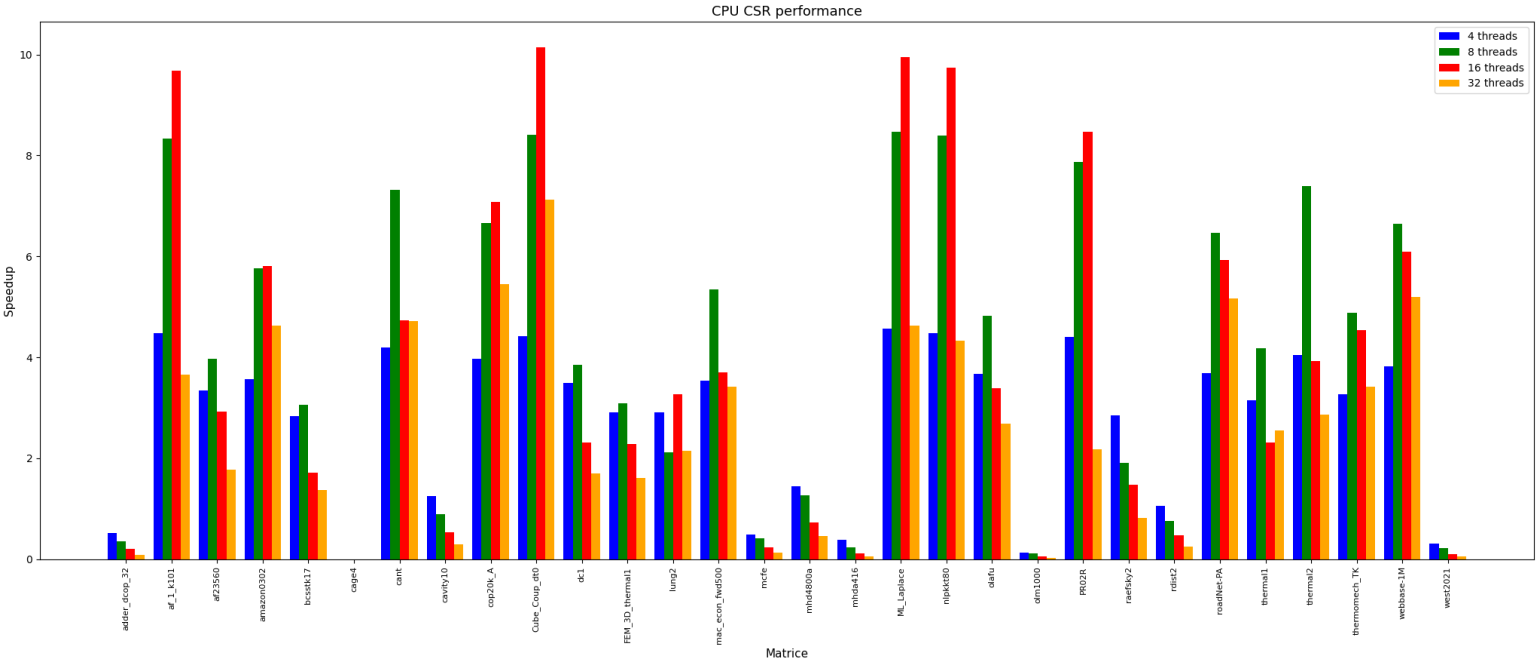

Figure 3: Speedup calcolo su CPU con formato CSR

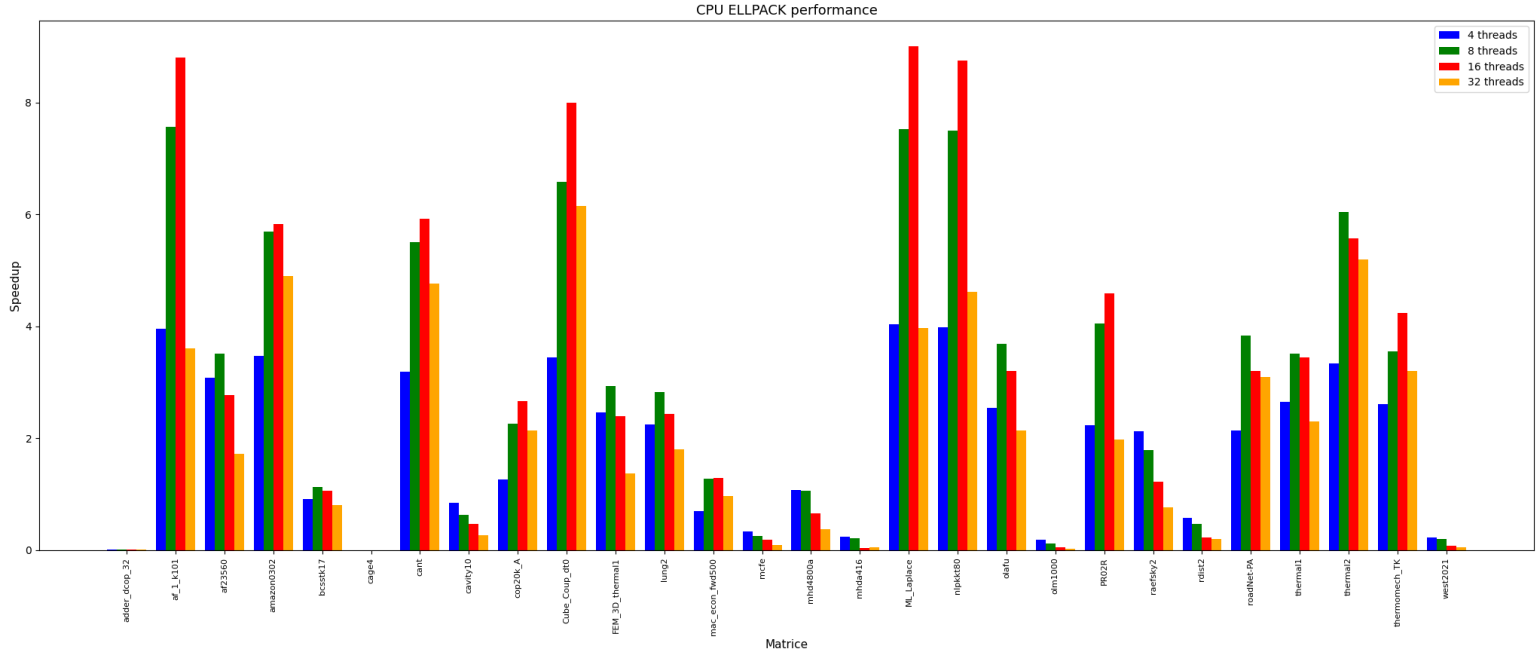

Figure 4: Speedup calcolo su CPU con formato ELLPACK

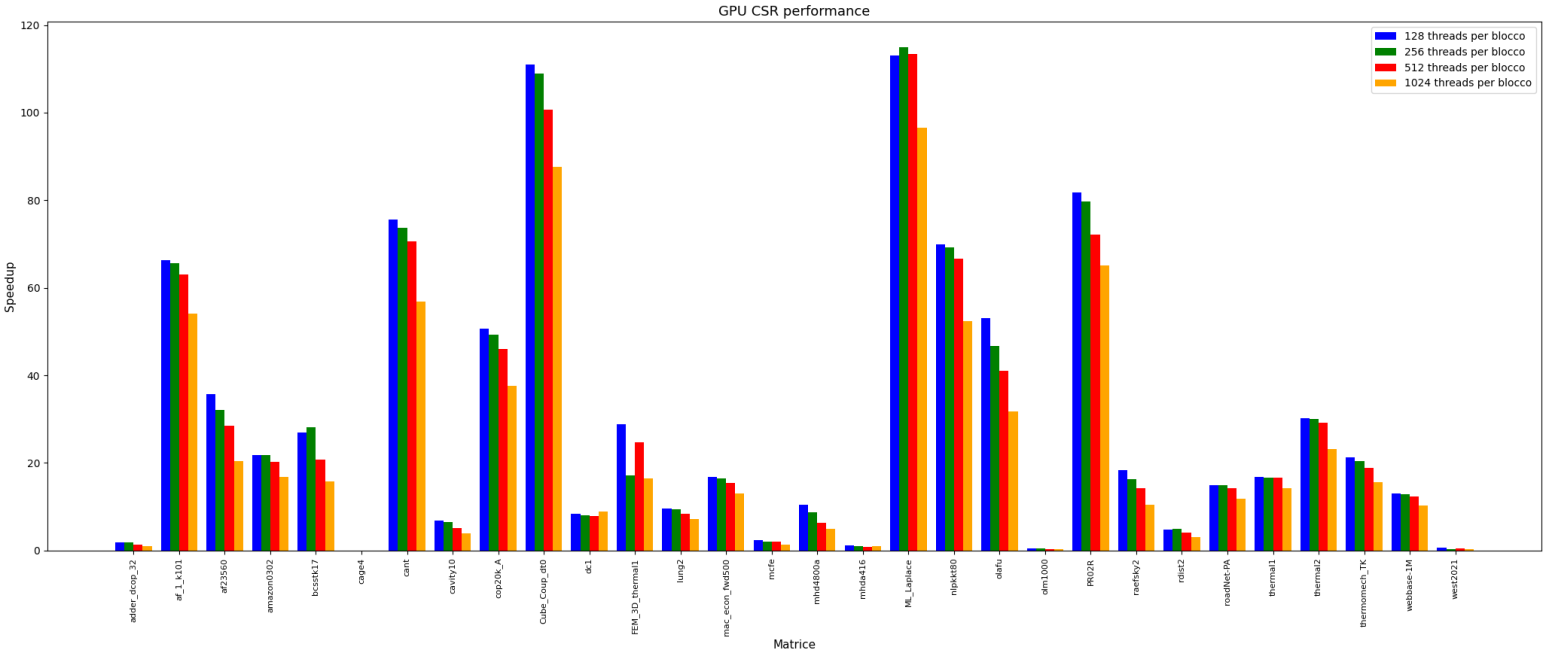

Figure 5: Speedup calcolo su GPU con formato CSR

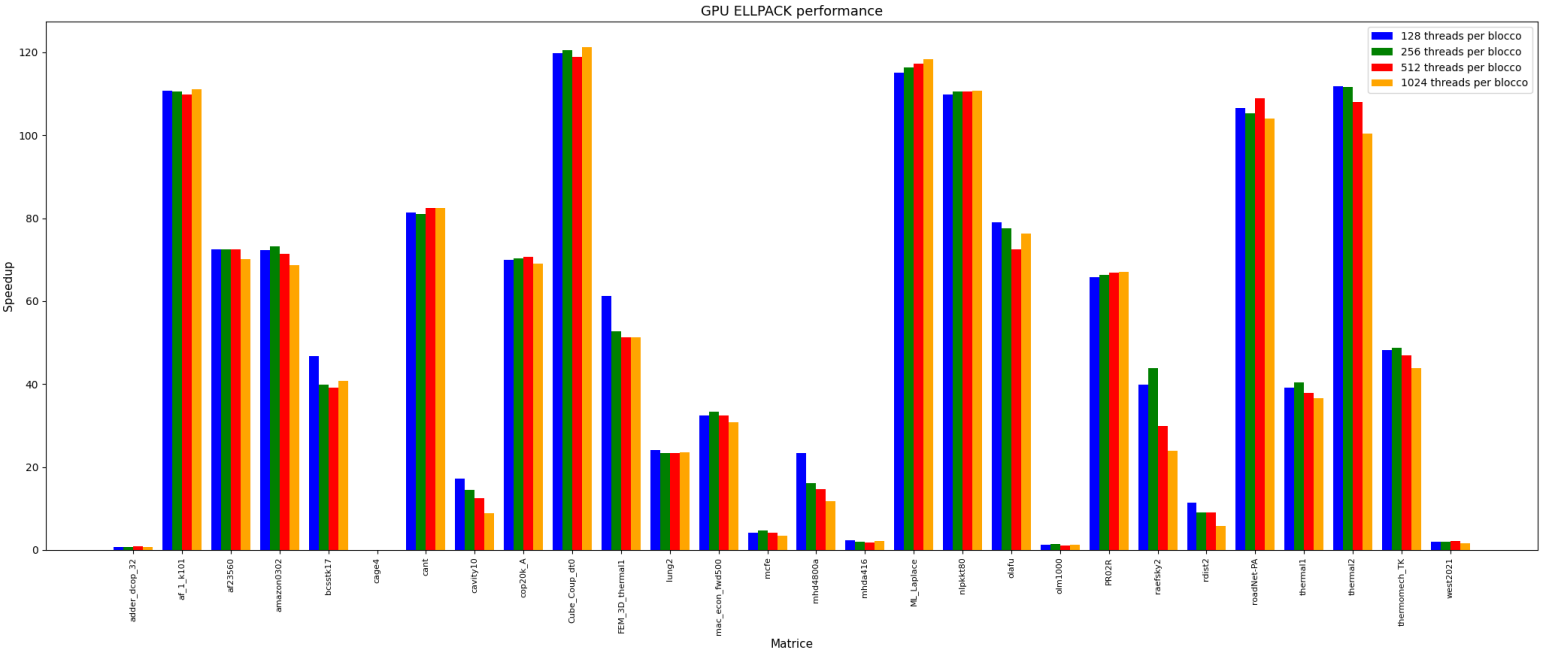

Figure 6: Speedup calcolo su GPU con formato ELLPACK

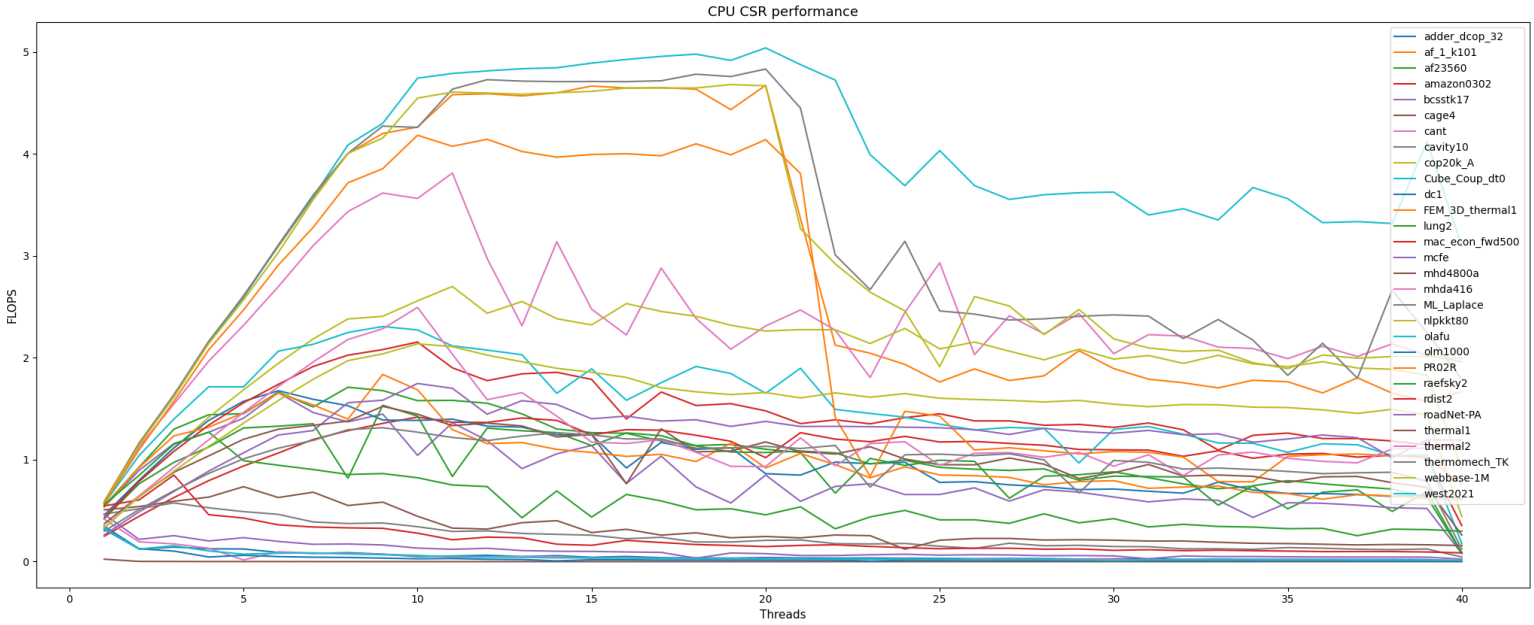

Figure 7: GFLOPS al variare dei threads nel calcolo su CPU con formato CSR

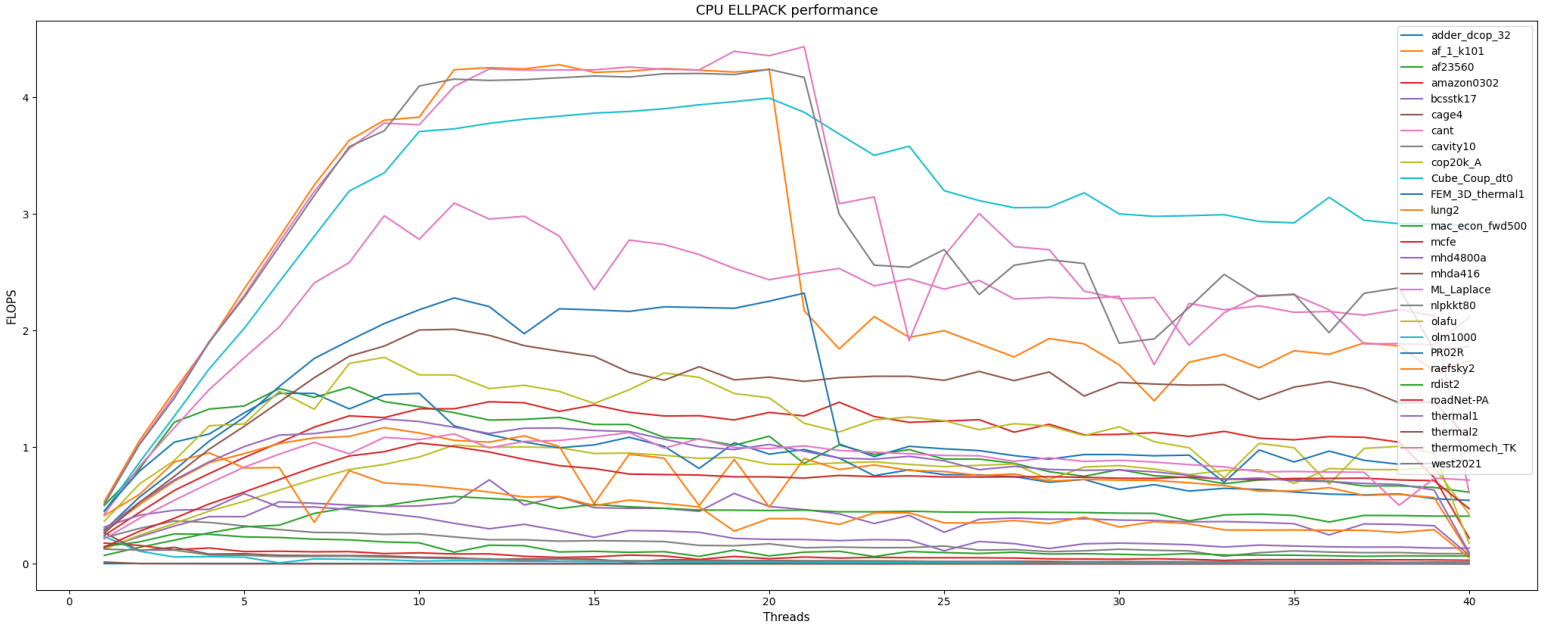

Figure 8: GFLOPS al variare dei threads nel calcolo su CPU con formato ELLPACK

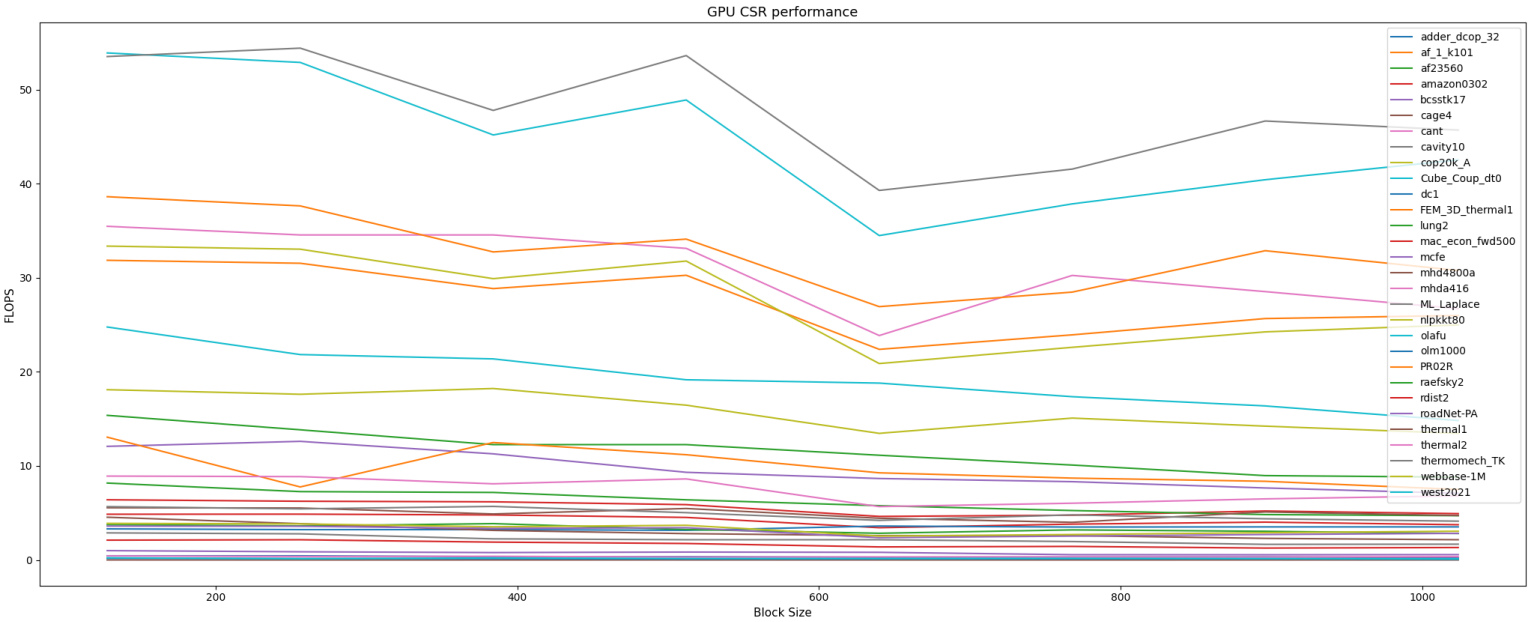

Figure 9: GFLOPS al variare della dimensione del blocco nel calcolo su GPU con formato CSR

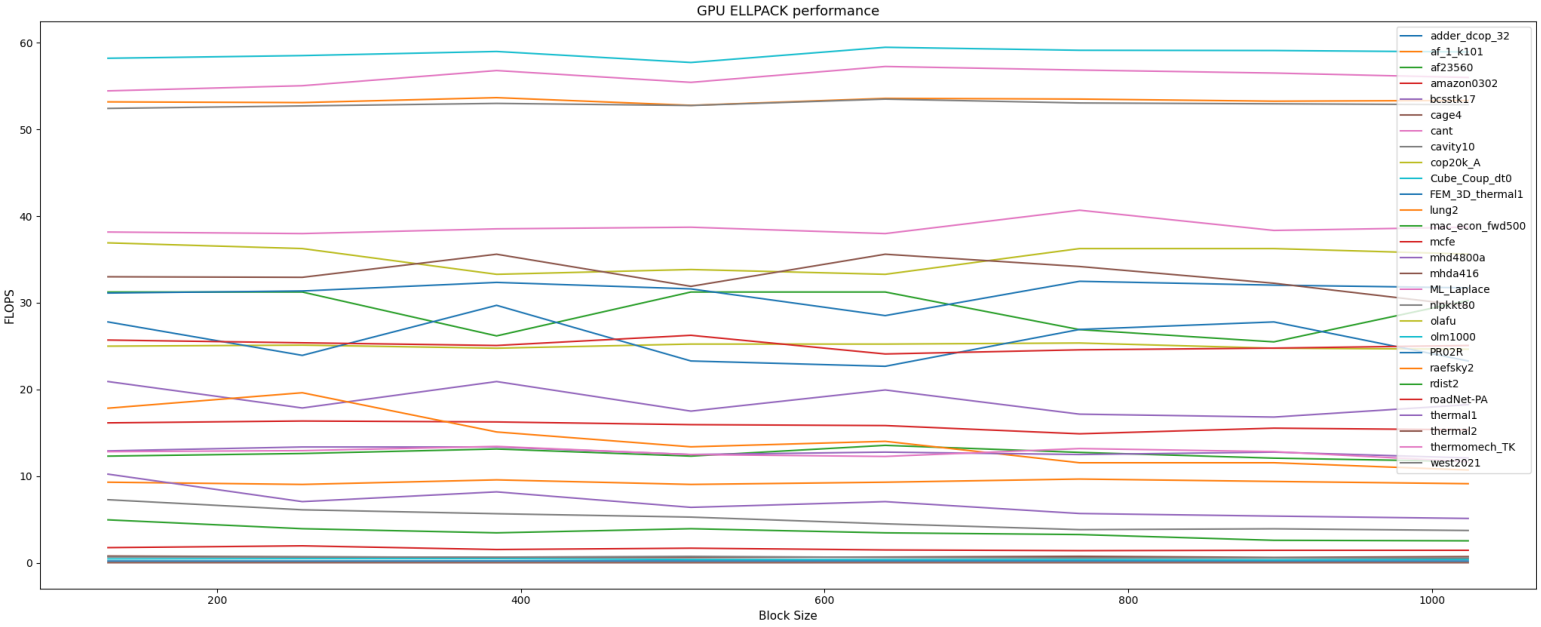

Figure 10: GFLOPS al variare della dimensione del blocco nel calcolo su GPU con formato ELLPACK