ParCFD2022
33rd International Conference on Parallel Computational Fluid Dynamics
May 25-27 2022, Alba Italy

ON THE BENEFITS AND APPLICATIONS OF SPARSE MATRIX-MATRIX PRODUCT ON VARIOUS PARALLEL ARCHITECTURES

Xavier Álvarez-Farré[1], Àdel Alsalti-Baldellou[1,2], Andrey Gorobets[3], Assensi Oliva[1] and F. Xavier Trias[1]

[1] Heat and Mass Transfer Technological Center, Technical University of Catalonia,
Carrer Colom 11, 08222 Terrassa (Barcelona), Spain.
e-mail: {xavier.alvarez.farre, adel.alsalti, asensio.oliva, francesc.xavier.trias}@upc.edu
[2] TermoFluids S.L. (https://www.termofluids.com), Sabadell (Barcelona), Spain.
[3] Keldysh Institute of Applied Mathematics, Russian Academy of Sciences,
Miusskaya Sq. 4, 125047 Moscow, Russia.
e-mail: andrey.gorobets@gmail.com

Key words: Sparse linear algebra, SpMM, Parallel methodology, Parallel CFD applications

Abstract. Sparse matrix-vector multiplications are essential in the numerical resolution of partial differential equations. However, the sparse matrix-vector product is a strongly memory-bound kernel that suffers from data starving. This work discusses the benefits and applications of the sparse matrix-matrix product, which simultaneously multiplies a sparse matrix by a set of vectors (*i.e.*, a dense matrix). Architecture-specific implementation details and performance analysis will be presented for different parallel architectures.

1 INTRODUCTION

Large sparse matrices often appear in diverse areas of computational physics and data science, more precisely in the numerical resolution of partial differential equations. The sparse pattern of these matrices (*i.e.*, the number and distribution of non-zero coefficients) depends on the spatial discretization of the computational domain and the numerical method employed. Hence, sparse matrix multiplications are widespread operations amongst the scientific computing community and receive a great deal of attention.

Sparse matrix-vector product (`SpMV`) is the most computationally expensive routine in many large-scale simulations relying on iterative methods. Namely, it is a strongly memory-bound kernel with a very low arithmetic intensity (AI), which is the ratio of computing work in floating-point operations (flop) to memory traffic in bytes (its value is around 1:8 flop per byte), and requires irregular memory accessing to the input vector harming the memory access efficiency. Significant effort is devoted to studying and optimizing `SpMV` for different applications and state-of-the-art computing environments. The introduction of the graphics processing units (GPUs) into high-performance computing (HPC) systems motivated the research of new sparse matrix storage formats and `SpMV` implementations, as reviewed by Filippone et al. in [1]. The continuous evolution of central processing units (CPUs) also motivates the research for efficient `SpMV` kernels on such architectures [2, 3]. However, the AI still limits all these efforts since the maximum achievable performance is $\beta \times \text{AI}$, where β is the processor's memory bandwidth [4].

2 OVERVIEW OF SPMM

In some cases, a sparse matrix is to be multiplied by a set of vectors. Consider the following expression:

$$\mathbf{y} = (diag(\mathbf{c}) \otimes \mathsf{A})\mathbf{x}, \tag{1}$$

where $\mathsf{A} \in \mathbb{R}^{m \times n}$ is a sparse matrix, $\mathbf{c} \in \mathbb{R}^{K}$, and $\mathbf{y} \in \mathbb{R}^{Km}$ and $\mathbf{x} \in \mathbb{R}^{Kn}$ are sets of vectors with K elements each.

Such formulation applies to several scenarios in numerical algorithms, allowing the use of the sparse matrix-matrix product (`SpMM`) kernel (Algorithm 1). Examples are symmetric grids [5], parallel in time methods [6], multiple transport equations or multiple parameter simulations [7], among others. Moreover, we will explore the nested combination of this approaches in a single framework to maximize the total number of components.

Algorithm 1 `SpMM` implementation using the standard CSR matrix format.

```
Require: A, x, c
Ensure: y
1: for $i \leftarrow 1$ to $m$ do
2:     sum $\leftarrow zeros(K)$
3:     for $j \leftarrow$ A.ptr[$i$] to A.ptr[$i+1$] do
4:         for $k \leftarrow 1$ to $K$ do
5:             sum[$k$] $\leftarrow$ sum[$k$] $+$ A.val[$j$] $\cdot$ x[A.idx[$j$]][$k$]
6:     for $k \leftarrow 1$ to $K$ do
7:         y[$i$][$k$] $\leftarrow$ c[$k$] $\cdot$ sum[$k$]
```

According to Algorithm 1, and considering double-precision, standard CSR sparse matrix format, and ideal temporal locality, the AI of the `SpMM` reads:

$$\text{AI}_{\texttt{SpMM}}(K) = \frac{(2\text{nnz}(A)+1)\cdot K}{8\text{nnz}(A) + 4\text{nnz}(A) + 4(m+1) + (8m+8n+8)\cdot K}. \tag{2}$$

where $\text{nnz}(A)$, m and n are the number of non-zero elements, rows and columns in the matrix, respectively, and K is the number of vectors. Consequently, the maximum speedup achievable by replacing K recursive `SpMV` calls with a single `SpMM` equals $\text{AI}_{\texttt{SpMM}}(K)/\text{AI}_{\texttt{SpMM}}(1)$. This upper-bound is plotted in Figure 1. The lower-bound is also given considering zero temporal locality (*i.e.*, accounting for the total number of memory accesses to the input vector, $8nnz(A)K$, instead of $8nK$). It is noteworthy that, being the upper-bound proportional to the average number of non-zeros per row, $\text{nnz}(A)/m$, the use of high order schemes may strengthen the benefits of the `SpMM`.

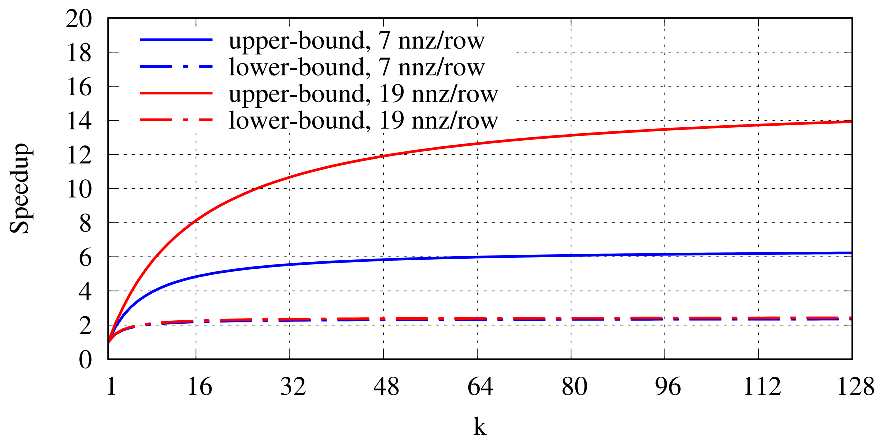

Figure 1: Different data layouts for sets of vectors.

3 PARALLEL PERFORMANCE STUDY ON VARIOUS ARCHITECTURES

At the conference, architecture-specific implementation details and performance analysis will be presented for different parallel architectures. Special focus will be given to the results for different vector data layouts (Figure 2), which have an enormous impact depending on the architecture. Namely, this affects not only the performance of the kernel but also the data exchanges protocol.

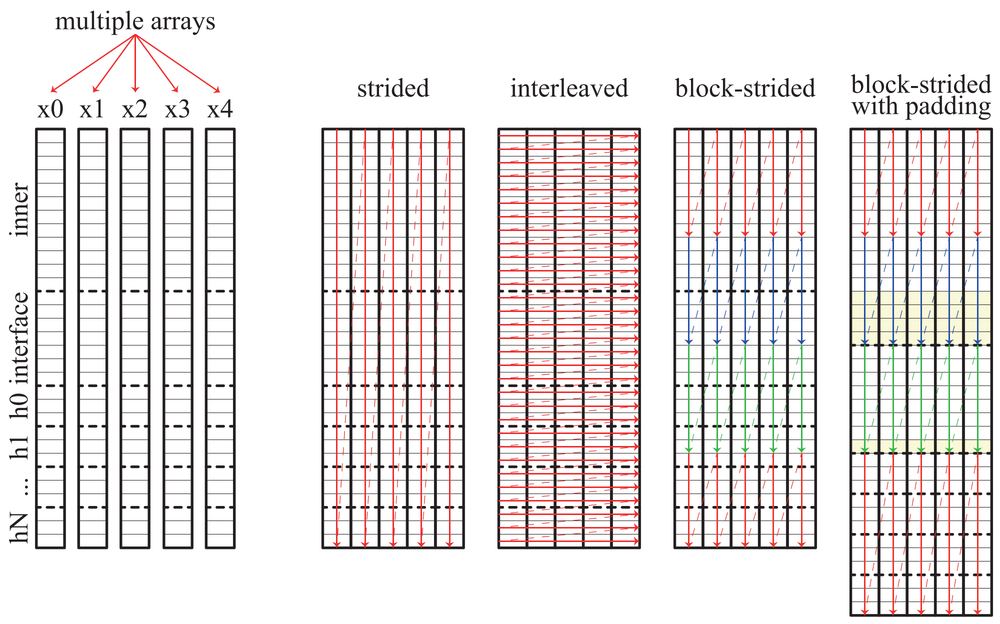

Figure 2: Block vectors ordering.

ACKNOWLEDGEMENTS

The work of X.Á.F, À.A.B, A.O., and F.X.T. has been financially supported by the following R+D projects: RETOtwin (PDC2021-120970-I00), given by MCIN/AEI/10.13039/501100011033 and European Union Next Generation EU/PRTR, FusionCAT (001-P-001722), given by *Generalitat de Catalunya* RIS3CAT-FEDER. À.A.B has also been supported by the predoctoral grants DIN2018-010061 and 2019-DI-90, given by MCIN/AEI/10.13039/501100011033 and the Catalan Agency for Management of University and Research Grants (AGAUR), respectively. The work of A. G. has been funded by the Russian Science Foundation, project 19-11-00299. The studies of this work have been carried out using computational resources of the Barcelona Supercomputing Center (IM-2020-3-0030 and IM-2022-1-0015). The authors thankfully acknowledge these institutions for the compute time and technical support.

REFERENCES

[1] S. Filippone, V. Cardellini, D. Barbieri, and A. Fanfarillo, "Sparse matrix-vector multiplication on GPGPUs," *ACM Trans. Math. Softw.*, vol. 43, jan 2017.

[2] X. Liu, M. Smelyanskiy, E. Chow, and P. Dubey, "Efficient sparse matrix-vector multiplication on x86-based many-core processors," in *Proceedings of the 27th International ACM Conference on International Conference on Supercomputing*, ICS '13, (New York, NY, USA), pp. 273–282, Association for Computing Machinery, June 2013.

[3] X. Álvarez-Farré, A. Gorobets, F. X. Trias, and A. Oliva, "NUMA-aware strategies for the heterogeneous execution of SpMV on modern supercomputers," in *World Congress in Computational Mechanics and ECCOMAS Congress*, January 2021.

[4] S. Williams, A. Waterman, and D. Patterson, "Roofline: An insightful visual performance model for multicore architectures," *Communications of the ACM*, vol. 52, no. 4, pp. 65–76, 2009.

[5] X. Álvarez-Farré, À. Alsalti-Baldellou, A. Gorobets, A. Oliva, and F. X. Trias, "Enabling larger and faster simulations from mesh symmetries," in *2nd High-Fidelity Industrial LES/DNS Symposium*, (Toulouse), September 2021.

[6] B. I. Krasnopolsky, "An approach for accelerating incompressible turbulent flow simulations based on simultaneous modelling of multiple ensembles," *Computer Physics Communications*, vol. 229, pp. 8–19, 2018.

[7] S. Imamura, K. Ono, and M. Yokokawa, "Iterative-method performance evaluation for multiple vectors associated with a large-scale sparse matrix," *International Journal of Computational Fluid Dynamics*, vol. 30, no. 6, pp. 395–401, 2016.