Efficient Algebraic Multigrid Preconditioners on Clusters of GPUs

Ambra Abdullahi Hassan[1], Valeria Cardellini[1], Pasqua D'Ambra[2], Daniela di Serafino[3], and Salvatore Filippone[4]

[1] Università degli Studi di Roma "Tor Vergata", Roma, Italy
{ambra.abdullahi,cardellini}@uniroma2.it
[2] Istituto per le Applicazioni del Calcolo "Mauro Picone", CNR, Napoli, Italy
pasqua.dambra@cnr.it
[3] Università degli Studi della Campania "Luigi Vanvitelli", Caserta, Italy
daniela.diserafino@unicampania.it
[4] Cranfield University, Cranfield, UK
salvatore.filippone@cranfield.ac.uk

Abstract Many scientific applications require the solution of large and sparse linear systems of equations using iterative methods, this operation accounting for a large percentage of the computing time. In these cases, the choice of the preconditioner is crucial for the convergence of the iterative method. Given the broad range of applications, great effort has been put in the development of efficient preconditioners ensuring algorithmic scalability: in this sense, multigrid methods have been proved to be particularly promising. Additionally, the advent of GPUs, now found in many of the fastest supercomputers, poses the problem of implementing efficiently these algorithms on highly parallel architectures; this is made more difficult by the fact that the solution of sparse triangular systems, a common kernel in many types of preconditioners, is extremely inefficient on GPUs.

In this paper, we use the PSBLAS and MLD2P4 libraries to explore various issues that affect the efficiency of multilevel preconditioners on GPUs, both in terms of execution speed and exploitation of computational cores, as well as the algorithmic efficiency in guaranteeing convergence to solution in a number of iterations independent of the parallelism degree. We investigate these issues in the context of linear systems arising from groundwater modeling application of the filtration of 3D incompressible single-phase flows through porous media.

1 Introduction

2 Multilevel preconditioner

Multilevel preconditioner e nuclei computazionali dellâĂŹalgoritmo di applicazione.

3 Sparse linear algebra on GPUs

General Purpose Graphics Processing Units (GPGPUs) [2] are today an established and attractive choice in the world of scientific computing, found in many among the fastest supercomputers on the Top 500 list, and offered in standard Cloud infrastructure services, e.g. in Amazon EC2. It is therefore not surprising that they are at the focus of many research efforts revolving around the efficient implementation of scientific software, including solvers for sparse linear system of equations.

Many applications use iterative methods to solve sparse linear systems; the most popular ones are those based on Krylov subspace projection methods [3].

From a software point of view, all Krylov methods employ the matrix A to perform matrix-vector products $y \leftarrow Ax$, whose efficient implementation provides significant challenges on all modern computer architectures. The SpMV kernel is well-known to be a memory bounded application; and its bandwidth usage is strongly dependent on both the input matrix and on the underlying computing platform(s). For a detailed overview of the implementation issues and a survey of available techniques for this kernel on GPUs, see [1].

For most real-world problems,it is necessary to combine the Krylov solvers with *preconditioners*. A preconditioner is a transformation applied to the coefficient matrix such that the resulting linear system has better convergence properties. Examples of popular preconditioners include Jacobi and Gauss-Seidel iterations, Block Jacobi and Additive Schwarz coupled with incomplete factorizations, and Algebraic Multigrid.

To implement a linear system solver on a parallel computing architecture we typically use a *domain decomposition* approach, in which subsets of the computational domain are assigned to different computing nodes. When considering

4 Results

Distribuzione dati fatta tenendo conto della geometria (3D) del problema e giustificandola rispetto al vantaggio che fornisce in termini di rapporto calcolo/comunicazione

Acknowledgements

References

1. Filippone, S., Cardellini, V., Barbieri, D., Fanfarillo, A.: Sparse matrix-vector multiplication on gpgpus. ACM Trans. Math. Softw. 43(4), 30:1–30:49 (Jan 2017), http://doi.acm.org/10.1145/3017994
2. Luebke, D., Harris, M., Govindaraju, N., Lefohn, A., Houston, M., Owens, J., Segal, M., Papakipos, M., Buck, I.: GPGPU: general-purpose computation on graphics hardware. In: Proc. of 2006 ACM/IEEE Conf. on Supercomputing. SC '06 (2006)
3. Saad, Y.: Iterative Methods for Sparse Linear Systems. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2nd edn. (2003)