From hybrid architectures to hybrid solvers

Pierre Ramet

Abstract for Workshop Celebrating 40 Years of Nested Dissection - 2013

From hybrid architectures to hybrid solvers

HiePACS team
Inria, Université de Bordeaux and CNRS joint project-team
(https://team.inria.fr/hiepacs/)

Solving large sparse systems $Ax = b$ of linear equations is a crucial and time-consuming step, arising in many scientific and engineering applications. Consequently, many parallel techniques for sparse matrix solution have been studied, designed and implemented based on factorization or hybrid iterative-direct approaches. In this context, graph partitioning and nested dissection ideas have played a crucial role.

Solving a sparse linear system by a direct method is generally a highly irregular problem that induces some challenging algorithmic problems and requires a sophisticated implementation scheme in order to fully exploit the capabilities of modern hierachical supercomputers. We focused first on the block partitioning and scheduling problem for high performance sparse parallel supernodal factorization with static pivoting for large sparse symmetric systems. Our strategy is suitable for non-symmetric sparse matrices with symmetric pattern, and for general distributed heterogeneous architectures the computation and communication performance of which are predictable in advance. This has led to software developments of the PaStiX solver (http://pastix.gforge.inria.fr/) [5].

New supercomputers incorporate many microprocessors which include themselves more and more computational cores. These new architectures induce strongly hierarchical topologies. In the context of distributed NUMA architectures, we studied optimization strategies to improve the scheduling of communications, threads and I/O. We have developed dynamic scheduling designed for NUMA architectures in the PaStiX solver. We were also interested in the dynamic adaptation of the computation grain to use efficiently multi-core architectures and shared memory. Experiments on several numerical test cases have been performed to prove the efficiency of the approach on different architectures [4].

We recently studied the replacement of this internal highly integrated scheduling strategy by two generic runtime frameworks namely PARSEC and StarPU. Those runtimes give the opportunity to execute the factorization tasks graph on emerging computers equipped with accelerators. A comparative study of the performances of the supernodal solver with three different schedulers is performed on manycore architectures and the improvements obtained with GPU accelerators will be presented. These results demonstrate that these DAG runtimes provide uniform programming interfaces to obtain portable high performance on different architectures on irregular problems as sparse direct factorizations [3, 4].

In addition to the activities on direct solvers, we also studied some robust preconditioning algorithms for iterative methods. The goal of these studies is to overcome the huge memory consumption inherent to the direct solvers in order to solve 3D problems of huge size (several million of unknowns). Our studies focused on the building of generic parallel preconditioners based on ILU factorizations. The classical ILU preconditioners use scalar algorithms that do not exploit well CPU power and are difficult to parallelize. This preconditioning approach is to define an adaptive blockwise incomplete factorization that is much more accurate (and numerically more robust) than the scalar incomplete factorizations commonly used to precondition Krylov subspace iterative solvers. Such incomplete factorization can take advantage of the latest breakthroughs in sparse direct methods and particularly should be very competitive in CPU time (effective power used from processors and good scalability) while avoiding the memory limitation encountered by direct methods on large 3D problems. In this approach, we focused on the critical problem to find approximate supernodes of ILU(k) factorizations. The problem is to find a coarser block structure of the incomplete factors [6].

We also investigated direct-iterative hybrid solvers that attempt to fully benefit from the robustness and effectivenes of sparse direct solvers and flexibility parallel implementation of Krylov subspace methods. We design two hybrid methods based on a Schur complement approach. The first technique uses the HID (hierarchical interface structure) ordering [7]. The principle is to build a decomposition of the adjacency matrix of the system into a set of small sub-domains (the typical size of a sub-domain is around a few hundreds or thousand nodes) with overlap. We build this decomposition from the nested dissection separator tree obtained using a sparse matrix reordering. Thus, at a certain level of the separator tree, the sub-trees are considered as the interior of the sub-domains and the union of the separators in the upper part of the elimination tree constitutes the interface between the sub-domains. The interior of these sub-domains are treated by a direct method. Solving the whole system is then equivalent to solve the Schur complement system on the interface between the sub-domains which has a much smaller dimension. We propose several algorithmic variants to iteratively solve the Schur complement system that can be adapted to the problem or target architecture features. This has led to software developments of the HIPS solver (http://hips.gforge.inria.fr/) [8].

The second approach is based on an algebraic domain decompostion method built from a partitioning via nested dissection of the adjacency graph associated with the sparse matrix. The internal variables are eliminated using a sparse, possibly parallel, direct solver and the resulting Schur complement system is solved using a preconditioned algebraic additive Schwarz [1, 2, 9] Krylov solver. This leads to software developments of the MaPHyS solver.

The main goal of this presentation will be to give an overview of the continuum between these various algorithmic approaches and to present the improvements of the algorithms and of the associated parallel implementations in a manycore context. Numerical experiments on large irregular real-life problems will illustrate this work.

References

[1] Agullo E., Giraud L., Guermouche A., and Roman J. Parallel hierarchical hybrid linear solvers for emerging computing platforms. *Compte Rendu de l'Académie des Sciences - Mécanique*, 339(2-3):96–105, 2011.

[2] Agullo E., Giraud L, Guermouche A., Haidar A. and Roman J. Parallel algebraic domain decomposition solver for the solution of augmented systems. *Advances in Engineering Software*, Elsevier, 2012.

[3] Bosilca, G., Faverge, M., Lacoste, X., Yamazaki, I., Ramet, P. Toward a supernodal sparse direct solver over DAG runtimes. In: Proceedings of PMAA'2012, London (2012)

[4] Faverge, M., Ramet, P. Fine Grain Scheduling for Sparse Solver on Manycore Architectures. In: 15th SIAM Conference on Parallel Processing for Scientific Computing, Savannah (2012)

[5] Hénon, P., Ramet, P., Roman, J. PaStiX: A High-Performance Parallel Direct Solver for Sparse Symmetric Definite Systems. Parallel Computing **28**(2) (2002) 301–321

[6] Hénon, P., Ramet, P., Roman, J. On finding approximate supernodes for an efficient ILU(k) factorization. Parallel Computing **34**(2) (2008) 345–362

[7] Hénon P. and Saad Y. A Parallel Multistage ILU Factorization Based on a Hierarchical Graph Decomposition. SIAM Journal on Scientific Computing, 28(6):2266–2293, 2006.

[8] Gaidamour, J., Hénon, P. A parallel direct/iterative solver based on a Schur complement approach. In: IEEE 11th International Conference on Computational Science and Engineering, Sao Paulo Brésil (2008) page 98–105

[9] Giraud L., Haidar A., and Saad Y. Sparse approximations of the Schur complement for parallel algebraic hybrid linear solvers in 3D. *Numerical Mathematics: Theory, Methods and Applications*, 3(3):276–294, 2010.