Algorithms To Build Robust Hybrid Direct/Iterative Sparse Linear Solvers

P. Hénon
INRIA Bordeaux Sud-Ouest and LaBRI, France
J. Gaidamour
LaBRI, CNRS and INRIA Bordeaux, Sud-Ouest, France

Nowaday, three dimensional numerical simulations often require a tremendous amount of resource. On one hands, direct methods can be mandatory to solve very ill-conditioned systems. But for large 3D simulations, they are constrained by prohibitive memory requirements and they also need a high amount of floating point operations. Iterative methods on the other hand require much less memory and are more scalable in general. Methods such as algebraic multigrids or multilevel solvers are optimal for some problems: a linear complexity in term of memory and number of operations can be obtained on elliptic problems for example but they lack adaptability in the general case. We will present an hybrid solver that try to combine the assets of direct methods and iterative methods. More precisely, we will give some algorithms to build a scalable hybrid direct/iterative solver based on a Schur complement approach. Our method is based on a domain decomposition of the matrix adjacency graph. We define a special ordering and partitioning of the interface between the domain in order to compute a robust ILU preconditioner of the Schur complement.

Consider the matrix A of a linear system where we distinguish two different sets of unknowns B and C, then A can be factored in the following manner:

$$\begin{pmatrix} A_B & F \\ E & A_C \end{pmatrix} = \begin{pmatrix} L_B & \\ EU_B^{-1} & S \end{pmatrix} \times \begin{pmatrix} U_B & L_B^{-1}F \\ & I \end{pmatrix},$$

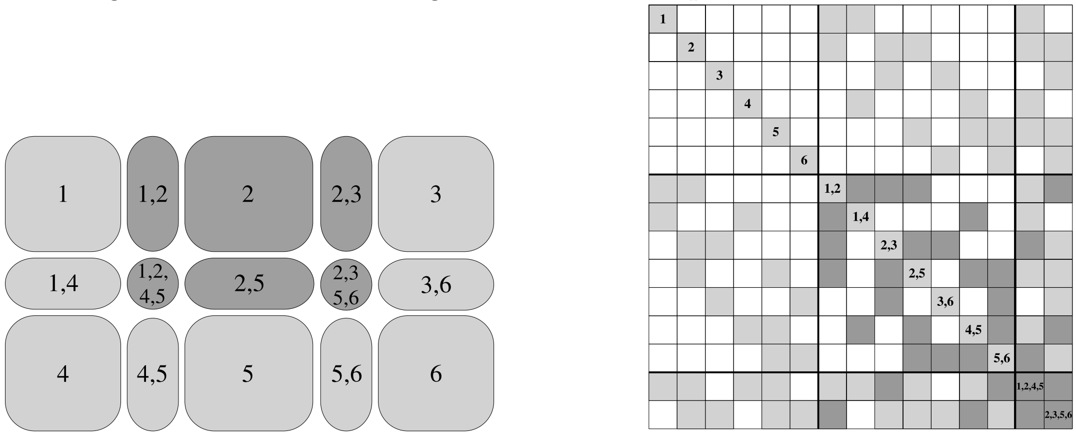
where $L_B.U_B$ is an exact factorization of A_B and S is the matrix of the Schur complement: $S = A_C - E.U_B^{-1}.L_B^{-1}.F$. In the following, we will denote $W = E.U_B^{-1}$ and $G = L_B^{-1}F$. Using this decomposition the linear system $A.x = r$ is equivalent to the system of equations:

$$\begin{aligned} A_B.z_B &= r_B \\ S.x_C &= r_C - E.z_B \\ A_B.x_B &= r_B - F.x_C \end{aligned}$$

where x_B , x_C , r_B and r_C are the part of x and r corresponding to the B and C sets of unknowns and z_B is a temporary vector. Thus, it can be solved in three steps: (1) $z_B = U_B^{-1}.(L_B^{-1}.r_B)$; (2) Use an iterative method to get x_C; (3) $x_B = U_B^{-1}.(L_B^{-1}.(r_B - F.x_C))$.

In our case, we reordered the matrix according to the Hierarchical Interface Decomposition (HID) we have described in the paper [1]: it consists in partitioning the set of unknowns of the interface into graph components named "connectors" that are grouped in "levels". A level of connectors plays the role of separators for the immediate inferior level. An illustration of this ordering and partitioning is given in figure 1: the bottom right part of the matrix correspond to the global Schur complement structure.

Figure 1: The HID ordering is illustrated on a partition of 6 subdomains.



Graph decomposition in 6 subdomains with overlap

The matrix is reordered and partitioned according to HID. The Schur complement structure corresponds to the right bottom part of the matrix.

The unknowns listed in the left part of the matrix (the B set) correspond to the interior vertices of the domain partition, and those in the right part of the matrix (the C set) correspond to the overlaping vertices reordered according to the HID. Hence, A_B is a block diagonal matrix where each block is a sparse matrix corresponding to the interior of a subdomain.

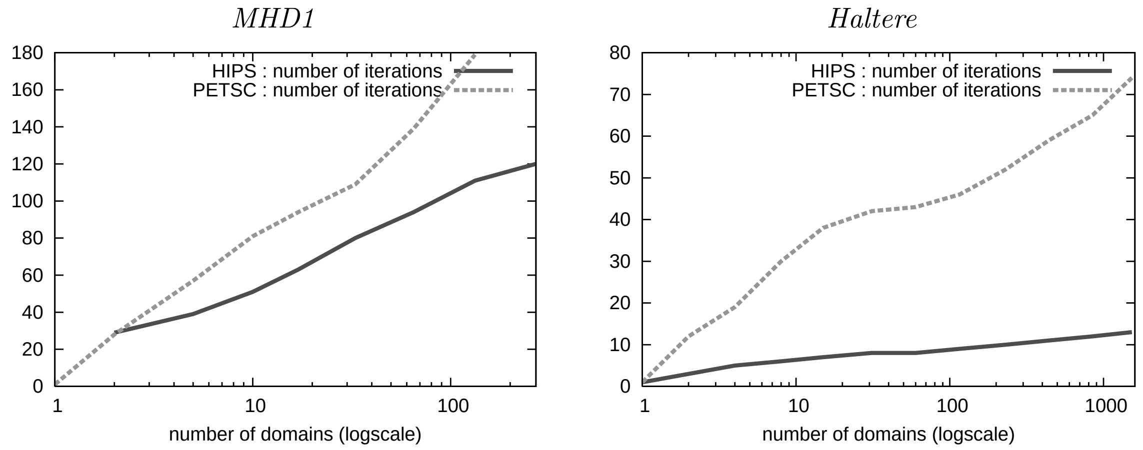
To obtain an incomplete factorization of S, we allow fill-in only in the Schur complement system: that is to say inside the block pattern illustrated in figure 1. This fill-in pattern ensures that the connectivity between the domains will remain the same in the factored Schur matrix than in the original A matrix. It is an important ingredient in our method: the incomplete Schur factorization is global but the fill-in pattern ensures that it can be viewed from the local part of the Schur complement inside each domain. Thus we keep a high degree of parallelism while being more robust than with a local preconditioning of each local Schur complement matrix (like we would do with an additive Schwarz preconditioner of the Schur complement).

Another important aspect in our method is to avoid as much as possible the high memory consumption required to compute the Schur complement preconditioner. To this end, we will present different strategies based on a fine coupling between supernodal algorithm (coming from direct factorization) and block ILUT algorithms (see [2]). In particular, we will show that the best combination of right and left looking algorithm depends on the approximation used to compute the Schur complement preconditioner.

The Figure 2 shows a comparison between our hybrid solver implemented in HIPS [3] and the additive Schwarz method implemented in PETSc (with MUMPS as local direct solver). The test cases are MHD1 (485, 597 unknowns and 24, 233, 141 non-zeros) an

Figure 2: Comparison between additive Schwarz method and HIPS on two irregular test cases (Haltere and MHD1)

Iterations :



unsymmetric real matrix (3D magneto-hydrodynamic problem) and Haltere (1, 288, 825 unknowns and 10, 476, 775 non-zeros) a symmetric complex matrix (3D electromagnetism problem).

As we can see on these cases, the number of iterations of HIPS is much more robust then the additive Schwarz method. These cases illustrate our choice of multiple subdomains by processors parallelization scheme: a better trade-off *memory / time* can be obtained with subdomains of small sizes in regards to the problem dimension. Consequently in many cases the corresponding number of subdomains is greater than the number of processors reasonably needed to solve the linear system.

In our talk, we will present more results and comparisons and also parallel results of our solver on thousand of processors. All the algorithms presented are implemented in HIPS that is freely downloadable on http://hips.gforge.inria.fr/

References

[1] P. Hénon, and Y. Saad. *A Parallel Multistage ILU Factorization based on a Hierarchical Graph Decomposition*. SIAM J. Sci. Comput., 28 (2006), pp. 2266–2293.

[2] J. Gaidamour and P. Hénon. *A parallel direct/iterative solver based on a Schur complement approach*. IEEE 11th International Conference on Computational Science and Engineering (2008), pp. 98105.

[3] Jérénon Gaidamour, Pascal Hénon, Yousef Saad. HIPS Web page, http://hips.gforge.inria.fr, 2008.