Nested dissection with balanced halo

Astrid Casadei, Pierre Ramet, Jean Roman

Abstract for CSC14: The Sixth SIAM Workshop on Combinatorial Scientific Computing

Nested dissection with balanced halo

Astrid Casadei[1,3], Pierre Ramet[1,3], and Jean Roman[1,2]

[1]INRIA Bordeaux Sud-Ouest & CNRS (LaBRI UMR 5800)
[2]Bordeaux Institute of Technology (IPB)
[3]Bordeaux University

Nested has been introduced by A. George in 1973 ([4]) and is now a well-known and very popular heuristic for sparse matrix ordering to reduce both the fill-in and the operation count during the numerical factorization. The basic standard idea is to build a "small separator C" associated with the original matrix in order to split the remaining vertices in two parts A and B of "almost equal size". The vertices of the separator C are ordered with the largest indices, and then the same method is applied recursively on the two subgraphs induced by A and B. This method has been implemented by graph partitioners such as METIS or SCOTCH whose main objectives are to minimize the size of the separator and to equilibrate the size of the two separated subgraphs. However, if we examine precisely the complexity analysis for the estimation of asymptotic bounds for fill-in or operation count when using Nested Dissection ordering, we can notice that the size of the halo of the separated subgraphs (set of external vertices adjacent to the subgraphs and previously ordered) play a crucial role in the asymptotic behavior achieved.

Moreover, this method based on a divide and conquer approach is also very well suited to maximize the number of independent computation tasks for parallel implementations. Then, by using the block data structure induced by the partition of separators of the original graph, very efficient parallel block solvers have been designed and implemented according to supernodal or multifrontal approaches. To name a few, one can cite MUMPS, PaStiX and SuperLU. Considering now hybrid methods mixing both direct and iterative solvers such as HIPS [3], MaPHYS [1, 5], PDSLIN [11] and ShyLU [10], obtaining a domain decomposition leading to a good balancing of both the size of domain interiors and the size of interfaces is a key point for load balancing and effciency in a parallel context.

For this purpose, we revisit the algorithm introduced by Lipton, Rose and Tarjan ([8]) in 1977 which performed the recursion in a different manner: at each level, we apply recursively the method to the subgraphs induced by $A \cup C$ on one hand, and $B \cup C$ on the other hand. In these subgraphs, vertices already ordered (and belonging to the previous separators) are the halo vertices and the partition of these overlapping subgraphs is performed with three objectives: balancing of the two new parts A' and B', balancing of the halo vertices in the parts A' and B' and minimizing the size of the separator C'.

Our work is implemented in the Scotch partitioner [9]. By default, Scotch strategy to find a separator is based on the multilevel method [6, 7]: the (sub-)graph is coarsened multiple times until it becomes small enough, then an algorithm called greedy graph growing is applied on the coarsest graph to partition it, and finally the graph is uncoarsened, projecting the coarse separator on finer graph and refining it using the Fiduccia-Mattheyses algorithm [2].

We have adapted the multilevel framework of Scotch in order to take into account the halo vertices from orginal to coarsest graph. Moreover, we have worked on two variants of greedy graph growing. The first one is called *double greedy graph growing*. Its principle is to pick two seed vertices as far as possible among the halo, and to make parts A and B grow from them, with attention paid to keep halo balanced

among the growing parts. The second approach, called *halo-first greedy graph growing*, works in a first stage on the sole halo graph, finding a separator of it. Once it is done, it defines the two halo parts A_h and B_h as two sets of seeds and make these sets grow in the whole graph to build A and B. Finally, we have also changed the Fiduccia-Mattheyses refinement algorithm in order to preserve the good balancing in the finer graphs. Our algorithms will be explained more deeply during the presentation, with detailed results.

In addition to this algorithmic work, and especially for the MaPHYS solver, a complexity analysis is currently carried on regarding the good asymptotic size for the subdomains (in which a sparse direct factorization is performed) in order to have the same asymptotic complexity bound for the computation of the preconditioner and for the iterative part. This is done by using the same theoretical framework used in [8] for general families of graph associated with 3D irregular problems for elliptic PDEs where the maximum number of iterations depends of the number of subdomains. The result is achieved by using balancing properties for the domain interior sizes and for the halo sizes as expected in the first work presented above.

References

[1] E. Agullo, L. Giraud, A. Guermouche, A. Haidar, and J. Roman. Parallel algebraic domain decomposition solver for the solution of augmented systems. *Advances in Engineering Software*, 60–61(0):23 – 30, 2013. CIVIL-COMP: Parallel, Distributed, Grid and Cloud Computing.

[2] C.M. Fiduccia and R. M. Mattheyses. A linear-time heuristic for improving network partitions. In *19th Conference on Design Automation*, pages 175–181, June 1982.

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

[4] A. George. Nested dissection of a regular finite element mesh. *SIAM Journal on Numerical Analysis*, 10(2):345–363, 1973.

[5] L. Giraud, A. Haidar, and Y. Saad. Sparse approximations of the Schur complement for parallel algebraic hybrid linear solvers in 3D. Rapport de recherche RR-7237, INRIA, March 2010.

[6] B. Hendrickson and R. Leland. A multi-level algorithm for partitioning graphs. In *Supercomputing, 1995. Proceedings of the IEEE/ACM SC95 Conference*, pages 28–28, 1995.

[7] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20(1):359–392, 1998.

[8] R. J. Lipton, Donald J. Rose, and Robert Endre Tarjan. Generalized nested dissection. *SIAM Journal on Numerical Analysis*, 16(2), 1979.

[9] F. Pellegrini and J. Roman. Scotch: A software package for static mapping by dual recursive bipartitioning of process and architecture graphs. In *High-Performance Computing and Networking*, volume 1067 of *Lecture Notes in Computer Science*, pages 493–498. Springer, 1996.

[10] S. Rajamanickam, E.G. Boman, and M.A. Heroux. ShyLU: A hybrid-hybrid solver for multicore platforms. In *Parallel Distributed Processing Symposium (IPDPS), 2012 IEEE 26th International*, pages 631–643, May 2012.

[11] Ichitaro Yamazaki and Xiaoye S. Li. On techniques to improve robustness and scalability of a parallel hybrid linear solver. In *High Performance Computing for Computational Science – VECPAR 2010*, volume 6449 of *Lecture Notes in Computer Science*, pages 421–434. Springer, 2011.