Flexible Task Allocation for the Memory Minimization of the Multifrontal Approach

Abdou Guermouche* and Jean-Yves L'Excellent†

Multifrontal methods [4, 5] are efficient direct methods to solve sparse systems of linear equations, in which the sparse matrix of the system is factored by performing a succession of partial factorizations of small dense matrices called *frontal matrices*. Each frontal matrix is associated to a node of a so-called *assembly tree*, which represents the dependencies of the tasks in the algorithm. Multifrontal methods work by assembling information (contribution blocks) from the frontal matrices of children to the frontal matrix of the parent. An individual assembly operation can only be performed after the child has been factored and after the frontal matrix of the parent has been allocated. But these scheduling constraints still leave significant freedom to choose the allocation schedule. In [8], Liu presents two extreme schedules and the corresponding reordering of children to reduce the active memory usage:

- the *classical allocation scheme* where the parent is only allocated after all children have been factored, and
- the *early allocation scheme* where the parent is allocated immediately after the first child has been factored.

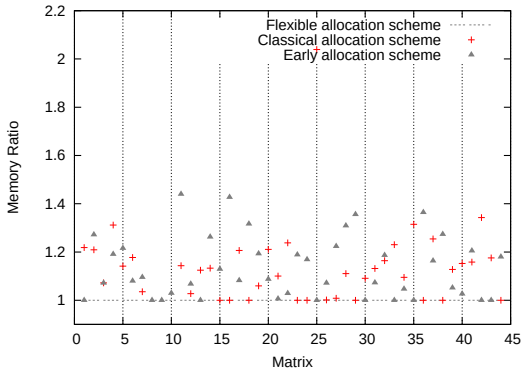
The objective of this talk is to show how to generalize these extreme schedules by allowing the parent to be allocated after any number of children have been factored (we will refer to this scheme as *flexible allocation scheme*). When the parent is allocated, all the contributions from its factored children are assembled and discarded. From that point on, each child that is factored is immediately assembled and its memory is released. Under these new assumptions, we need to find at each level of the tree (starting with leave nodes) an optimal schedule, consisting in:

- the order in which the children of a node should be processed, and
- the position where the parent node should be allocated.

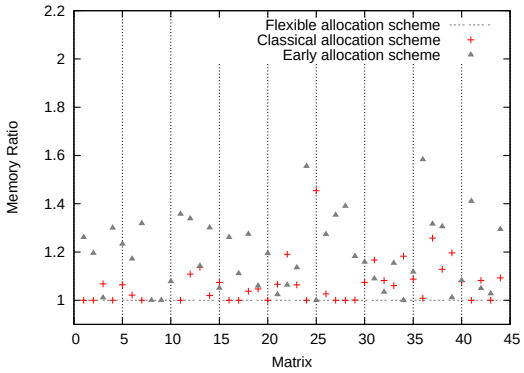
We propose algorithms to find such an optimal schedule so that either the maximum memory consumption (including the generated factors) is minimized, or the active memory (current frontal matrix and contribution blocks not yet assembled) is minimized. In Figure 1, we report the gains obtained when the new schedules are applied in the case of the active memory minimization with a large range of test problems (44 matrices from various sparse matrix collections). We used the software package `MUMPS` (MUltifrontal Massively Parallel Solver) [2], which implements a parallel multifrontal solver with threshold partial pivoting for both $\mathbf{LU}$ and $\mathbf{LDL}^T$ factorizations. We implemented the algorithms in the analysis phase of `MUMPS` and we compute statically the memory occupation for the factorization (if no pivoting occurs). In this simulator, the tree is traversed using a depth-first search scheme (like the one used in the

*INRIA/ENSEEIHT-IRIT, Toulouse, France.
†INRIA/LIP-ENS Lyon, Lyon, France.

factorization phase of a multifrontal method), and the schedules defined above are used to measure the memory peak. Note that in the results presented in Figure 1 we used two types of sparse matrix oderings, `AMD` (Approximate Minimum Degree) [1] and `METIS` [7].



(a) `AMD`. Gains relative to the *classical allocation scheme* are equal to 14.46, 10.10 and 14.5 for matrices 8, 9 and 10, respectively.



(b) `METIS`. Gains relative to the *classical allocation scheme* are equal to 27.18, 17.48 and 19.60 for matrices 8, 9 and 10, respectively.

Figure 1: Comparison of the active memory usage with different parent allocation schemes. For each scheme, the schedule is chosen to minimize the active memory usage. Memory is normalized with respect to the flexible allocation scheme.

These results show the potential and the interest of our approach to minimize the memory usage of the multifrontal method. We report in [6] more results with several variants of the proposed schedule.

References

[1] P. R. Amestoy, T. A. Davis, and I. S. Duff. An approximate minimum degree ordering algorithm. *SIAM Journal on Matrix Analysis and Applications*, 17:886–905, 1996.

[2] P. R. Amestoy, I. S. Duff, J. Koster, and J.-Y. L'Excellent. A fully asynchronous multifrontal solver using distributed dynamic scheduling. *SIAM Journal on Matrix Analysis and Applications*, 23(1):15–41, 2001.

[3] P. R. Amestoy, I. S. Duff, and J.-Y. L'Excellent. Multifrontal parallel distributed symmetric and unsymmetric solvers. *Comput. Methods Appl. Mech. Eng.*, 184:501–520, 2000.

[4] I. S. Duff and J. K. Reid. The multifrontal solution of indefinite sparse symmetric linear systems. *ACM Transactions on Mathematical Software*, 9:302–325, 1983.

[5] I. S. Duff and J. K. Reid. The multifrontal solution of unsymmetric sets of linear systems. *SIAM Journal on Scientific and Statistical Computing*, 5:633–641, 1984.

[6] A. Guermouche and J.-Y. L'Excellent. Optimal memory minimization algorithms for the multifrontal method. Technical Report RR5179, INRIA, 2004. Also LIP report RR2004-26.

[7] G. Karypis and V. Kumar. MeTiS – *A Software Package for Partitioning Unstructured Graphs, Partitioning Meshes, and Computing Fill-Reducing Orderings of Sparse Matrices – Version 4.0.* University of Minnesota, September 1998.

[8] J. W. H. Liu. On the storage requirement in the out-of-core multifrontal method for sparse factorization. *ACM Transactions on Mathematical Software*, 12:127–148, 1986.