REDUCING ELIMINATION TREE HEIGHT FOR UNSYMMETRIC MATRICES

ENVER KAYAASLAN† AND BORA UÇAR‡

Abstract. The elimination tree for unsymmetric matrices is a recent model playing important roles in sparse LU factorization. This tree captures the dependencies between the tasks of some well-known variants of sparse LU factorization. Therefore, the height of the elimination tree roughly corresponds to the critical path length of the task dependency graph in the corresponding parallel LU methods. We propose algorithms to symmetrically permute the rows and columns of a given unsymmetric matrix so that the height of the elimination tree is reduced, and thus a high degree of parallelism is exposed. The proposed algorithms are obtained by generalizing the most successful approaches used in sparse Cholesky factorization. We test the proposed algorithms on a set of real world matrices and report noticeable reduction in the elimination tree heights with respect to a possible exploitation of the state of the art tools used in Cholesky factorization.

1. Introduction. The standard elimination tree [15] has been used to expose parallelism in sparse Cholesky, LU, and QR factorizations [1, 3, 8, 11]. Roughly, a set of vertices without ancestor/descendant relations corresponds to a set of independent tasks that can be performed in parallel. Therefore, the total number of parallel steps, or the critical path length, is equal to the height of the tree on an unbounded number of processors [12]. Obtaining an elimination tree with the minimum height for a given matrix is NP-complete [14]. Therefore, heuristic approaches are used. One set of heuristic approaches is to content oneself with the graph partitioning based methods. These methods reduce some other important cost metrics in sparse Cholesky factorization, such as the fill-in and the operation count, while giving a shallow depth elimination tree [7]. When the matrix is unsymmetric, the elimination tree for LU factorization [4] would be useful to expose parallelism as well. In this respect, the height of the tree, again, corresponds to the number of parallel steps or the critical path length for certain factorization schemes. In this work, we develop heuristics to reduce the height of elimination trees for unsymmetric matrices. To the best of our knowledge no other work looked at this problem on its own.

Let $\mathbf{A}$ be a square matrix and let $\mathrm{G}(\mathbf{A}) = (\mathrm{V}(\mathbf{A}), \mathrm{E}(\mathbf{A}))$ be the standard directed graph model. The minimum hight of an elimination tree of $\mathbf{A}$ is equivalent to the graph theoretical notion of the cycle-rank of $\mathrm{G}(\mathbf{A})$. Gruber [6] shows that computing the cycle-rank is NP-complete, justifying the need for heuristics for large problems. One reasonable heuristic is to use a graph partitioning tool, such as MeTiS [9], on the symmetrized matrix (we present comparisons with this method). One can also use some local ordering heuristics [2]; but as their analogue for symmetric matrices, these are not expected to be very effective.

2. Methodology. We propose a recursive approach to reorder a given matrix so that the elimination tree is reduced. The main procedure takes an irreducible unsymmetric matrix $\mathbf{A}$ as its input, and produces a permutation yielding an upper bordered block diagonal form, as shown in (2.1)

$$\mathbf{A}_{\mathrm{BBT}} = \mathbf{P}\mathbf{A}\mathbf{P}^T = \begin{bmatrix} \mathrm{A}_{11} & \mathrm{A}_{12} & \dots & \mathrm{A}_{1\mathrm{K}} & \mathrm{A}_{1\mathrm{B}} \\ & \mathrm{A}_{22} & \dots & \mathrm{A}_{2\mathrm{K}} & \mathrm{A}_{2\mathrm{B}} \\ & & \ddots & \vdots & \vdots \\ & & & \mathrm{A}_{\mathrm{KK}} & \mathrm{A}_{\mathrm{KB}} \\ \mathrm{A}_{\mathrm{B}1} & \mathrm{A}_{\mathrm{B}2} & \dots & \mathrm{A}_{\mathrm{BK}} & \mathrm{A}_{\mathrm{BB}} \end{bmatrix}. \tag{2.1}$$

At each recursive call, the procedure partitions the current matrix into a 2×2 block structure where the off-diagonal blocks have only a few nonzeros. Then a covering for each off diagonal blocks is found and the better one is used to have a BBT form with two blocks. Then, the recursion is

†INRIA, France and LIP, ENS Lyon(enver.kayaaslan@ens-lyon.fr).

‡CNRS and Laboratoire de l'Informatique du Parallélisme, (UMR CNRS -ENS Lyon-INRIA-UCBL), Université de Lyon, 46, allée d'Italie, ENS Lyon, F-69364, Lyon Cedex 7, France, bora.ucar@ens-lyon.fr).

matrix	MeTiS	BBT-3	BBT-50	matrix	MeTiS	BBT-3	BBT-50
Averous/epb1	401	325	323	Hohn/fd18	422	307	311
Bai/rw5151	268	168	168	Hohn/sinc12	1836	1206	1181
Goodwin/goodwin	422	369	371	Hollinger/g7jac040	991	762	756
Graham/graham1	549	466	467	Lucifora/cell1	193	125	133
Grund/bayer02	198	123	119	Nasa/barth	142	99	102
Grund/bayer10	211	141	141	Nasa/barth4	133	80	83
Hamrle/Hamrle2	103	67	67	Nasa/barth5	185	117	103
Hohn/fd12	260	199	202	Shen/e40r0100	617	523	597
Hohn/fd15	381	250	251	TOKAMAK/utm5940	448	387	388

TABLE 3.1
Height of the elimination tree on a set of matrices.

applied to each diagonal block. The recursion stops when the current matrix has a size smaller than a parameter (we tested with 3 and 50). Then another heuristic based on feedback vertex sets is used to order the smallest matrices. The overall approach is the unsymmetric analog of pioneering work on extracting vertex separators from edge separators [13].

3. Results. We present results on matrices used in previous study, where the unsymmetric elimination trees were algorithmically studied [5]. Table 3.1 compares MeTiS and the proposed method with the stopping condition of the recursion being 3 and 50, which are shown as BBT-3 and BBT-50. In this table, BBT-3 and BBT-50 obtain tree heights whose geometric mean to the heights obtained by MeTiS is 0.71 and 0.72, where each run is the median of 5 runs (MeTiS has randomization inside).

Given this strikingly good results, we further performed tests with other data. We used the matrices (a subset from [10], where the matrices with a pattern symmetry of a most 0.90 are used, and only two matrices from each group is used). In this data set, the geometric mean of the heights of trees obtained by BBT-50 to MeTiS is found to be 0.91.

The poster presentation will include algorithmic details and further details.

REFERENCES

[1] E. AGULLO, A. BUTTARI, A. GUERMOUCHE, AND F. LOPEZ, *Multifrontal QR factorization for multicore architectures over runtime systems*, in Euro-Par 2013 Parallel Processing, F. Wolf, B. Mohr, and D. Mey, eds., vol. 8097 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2013, pp. 521–532.

[2] P. AMESTOY, X. LI, AND E. NG, *Diagonal Markowitz scheme with local symmetrization*, SIAM J. Matrix Anal. Appl., 29 (2007), pp. 228–244.

[3] P. R. AMESTOY, I. S. DUFF, AND J.-Y. L'EXCELLENT, *Multifrontal parallel distributed symmetric and unsymmetric solvers*, Computer methods in applied mechanics and engineering, 184 (2000), pp. 501–520.

[4] S. C. EISENSTAT AND J. W. H. LIU, *The theory of elimination trees for sparse unsymmetric matrices*, SIAM J. Matrix Anal. Appl., 26 (2005), pp. 686–705.

[5] S. C. EISENSTAT AND J. W. H. LIU, *Algorithmic aspects of elimination trees for sparse unsymmetric matrices*, SIAM J. Matrix Anal. Appl., 29 (2008), pp. 1363–1381.

[6] H. GRUBER, *Digraph complexity measures and applications in formal language theory*, Discrete Mathematics & Theoretical Computer Science, 14 (2012), pp. 189–204.

[7] A. GUERMOUCHE, J.-Y. L'EXCELLENT, AND G. UTARD, *Impact of reordering on the memory of a multifrontal solver*, Parallel Computing, 29 (2003), pp. 1191–1218.

[8] J. HOGG, J. REID, AND J. SCOTT, *Design of a multicore sparse Cholesky factorization using DAGs*, SIAM Journal on Scientific Computing, 32 (2010), pp. 3627–3649.

[9] G. KARYPIS AND V. KUMAR, *A software package for partitioning unstructured graphs, partitioning meshes, and computing fill-reducing orderings of sparse matrices*, University of Minnesota, Department of Computer Science and Engineering, Army HPC Research Center, Minneapolis, MN, (1998).

[10] K. KAYA AND B. UÇAR, *Constructing elimination trees for sparse unsymmetric matrices*, SIAM J. Matrix Anal. Appl., 34 (2013), pp. 345–354.

[11] J. W. LIU, *Computational models and task scheduling for parallel sparse Cholesky factorization*, Parallel Computing, 3 (1986), pp. 327–342.

[12] ———, *Reordering sparse matrices for parallel elimination*, Parallel Computing, 11 (1989), pp. 73 – 91.

[13] J. W. H. LIU, *A graph partitioning algorithm by node separators*, ACM Trans. Math. Softw., 15 (1989), pp. 198–219.

[14] A. POTHEN, *The complexity of optimal elimination trees*, Tech. Rep. CS-88-13, Pennsylvania State Univ., 1988.

[15] R. SCHREIBER, *A new implementation of sparse Gaussian elimination*, ACM Transactions on Mathematical Software, 8 (1982), pp. 256–276.