NLAFET

Parallel Numerical Linear Algebra
for Future Extreme Scale Systems

H2020–FETHPC–2014: GA 671633

D3.1

Theoretical bounds for communication in sparse operations

April 2017

Document information

Scheduled delivery	2017-04-30
Actual delivery	2017-04-28
Version	1.0
Responsible partner	INRIA

Dissemination level

PU — Public

Revision history

Date	Editor	Status	Ver.	Changes
2016-04-01	L. Grigori	Draft	0.1	Initial version of document produced
2016-04-24	L. Grigori	Final	1.0	Final version of revised document produced

Author(s)

Laura Grigori, INRIA

Internal reviewers

Florent Lopez, STFC
Iain Duff, STFC
Mawussi Zounoun, UNIMAN

Copyright

Acknowledgements

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under the grant agreement number 671633.

Table of Contents

List of Figures

1 Executive summary

One of the main challenges we face in high performance computing is the exponentially growing gap between the time required to perform floating point operations and the time required to transfer data. To address this challenge, our project focuses on designing algorithms that reduce drastically the communication cost of linear algebra operations, or even minimize it when possible.

In recent years, lower bounds on communication for dense linear algebra operations were derived [5, ?]. These bounds allowed the derivation of algorithms for dense linear algebra that minimize communication. However, few results exist in the case of sparse matrices.

In this document we discuss and review some of the results on the communication complexity of sparse matrix operations. We discuss first the communication complexity of the multiplication of two sparse matrices, and then we discuss the communication complexity of sparse matrix factorizations such as Cholesky or QR.

2 Introduction

The *Description of Action* document states for Deliverable 3.1:

> "*Theoretical bounds for communication in sparse operations.*
>
> Report on theoretical lower bounds for key sparse matrix operations such as matrix-matrix multiplication and factorization."

This deliverable is in the context of Task 3.1 (Lower Bounds on Communication for Sparse Matrices).

In this document we discuss and review several results on the communication complexity of sparse matrix operations. We present first the representation of the nonzero structure of a matrix by using graphs. In section 3 we consider the multiplication of two sparse matrices and we review results from [2, 3] which derive lower bounds on communication for matrices with random sparsity structure. In section 4 we consider direct methods of factorization of sparse matrices. We consider first matrices whose graphs have small separators. We review results from [15] on lower bounds on communication for the Cholesky factorization of a model problem. We then present bounds on communication for the QR factorization of a matrix whose graph has small separators. We then present results for matrices whose graphs don't have small separators.

2.1 Sparse matrices and graphs

The nonzero structure of a symmetric matrix can be represented by using an undirected graph. Given an $n \times n$ symmetric matrix A, the undirected graph $G(A) = (V, E)$ has a set V of n vertices and a set E of edges $\{i, j\}$, where i and j are two distinct vertices. There is a vertex i for each column/row of the matrix A and an edge $\{i, j\}$ for each nonzero element $A_{ij} = A_{ji}$. A path in a graph is a sequence of vertices which are connected.

We consider in the following an $n \times n$ matrix A which is symmetric and positive definite and whose structure can be represented by an undirected graph. As an example, the graph of the matrix A in equation (2.1) which is obtained from a finite difference operator on a two-dimensional (2D) regular grid using a five-point stencil is displayed at the left of Figure 1.

$$A = \begin{array}{c} \\ 1\\2\\3\\4\\5\\6\\7\\8\\9 \end{array} \begin{array}{c} \begin{array}{ccccccccc} 1&2&3&4&5&6&7&8&9 \end{array} \\ \left(\begin{array}{ccccccccc}
x & x & & x & & & & & \\
x & x & x & & x & & & & \\
 & x & x & & & x & & & \\
x & & & x & x & & x & & \\
 & x & & x & x & x & & x & \\
 & & x & & x & x & & & x \\
 & & & x & & & x & x & \\
 & & & & x & & x & x & x \\
 & & & & & x & & x & x
\end{array}\right) \end{array} \tag{2.1}$$

In this case, the Cholesky factorization $A = LL^T$ leads to a factor L that has more nonzeros than the matrix A. These new nonzero elements are referred to as fill-in elements, and the graph that contains them is referred to as the filled graph of A, $G^+(A) = (V, E^+)$. It has the same vertices as $G(A)$ and an edge $\{i, j\}$ if and only if there is a path from i to j in $G(A)$ whose intermediate nodes are numbered lower than $\min(i, j)$ [18, 19, 20]. If we ignore numerical cancellations, $G(L + L^T) = G^+(A)$. The elimination tree $T(A)$ [10, 21] has a node for each column of A and i is the parent of j if the first nonzero in column j of L is in row i. This tree can be computed directly from the nonzero structure of A. It provides information about the dependencies between column computations. If i is the parent of j in this tree, column j has to be factored before column i. Columns belonging to disjoint subtrees can be factored independently.

$$L + L^T = \begin{array}{c} \\ 1\\2\\3\\4\\5\\6\\7\\8\\9 \end{array} \begin{array}{c} \begin{array}{ccccccccc} 1&2&3&4&5&6&7&8&9 \end{array} \\ \left(\begin{array}{ccccccccc}
x & x & & x & & & & & \\
x & x & x & \color{red}{x} & x & & & & \\
 & x & x & \color{red}{x} & \color{red}{x} & x & & & \\
x & \color{red}{x} & \color{red}{x} & x & x & & x & & \\
 & x & \color{red}{x} & x & x & x & \color{red}{x} & x & \\
 & & x & & x & x & \color{red}{x} & \color{red}{x} & x \\
 & & & x & \color{red}{x} & \color{red}{x} & x & x & \\
 & & & & x & \color{red}{x} & x & x & x \\
 & & & & & x & & x & x
\end{array}\right) \end{array} \tag{2.2}$$

Nested dissection [12] is a global graph partitioning procedure that allows the matrix A to be reordered by identifying in the graph of A a small separator which divides the graph into two disjoint subgraphs. The process is then repeated recursively on the two

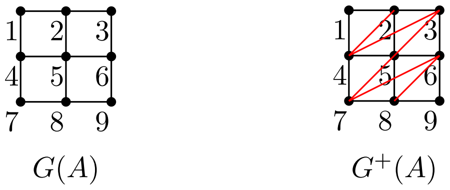

Figure 1: The graph $G(A)$ of matrix A from equation (2.1) and its filled graph $G^+(A)$.

subgraphs. By permuting the matrix A from equation (2.1) such that the columns/rows corresponding to the vertices of the separator are ordered last, followed by those of the two disjoint subgraphs, its Cholesky factors have the nonzero structure given in equation (2.3). The graph and the filled graph of the permuted matrix PAP^T are given in Figure 2. The elimination tree is presented on the right of this figure.

$$L + L^T = \begin{array}{c|cc|c|cc|c|ccc|} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 1 & x & & x & & & & x & & \\ 2 & & x & x & & & & & & x \\ \hline 3 & x & x & x & & & & x & x & x \\ \hline 4 & & & & x & & x & x & & \\ 5 & & & & & x & x & & & x \\ \hline 6 & & & & x & x & x & x & x & x \\ 7 & x & & x & x & & x & x & x & x \\ 8 & & & x & & & x & x & x & x \\ 9 & & x & x & & x & x & x & x & x \end{array} \tag{2.3}$$

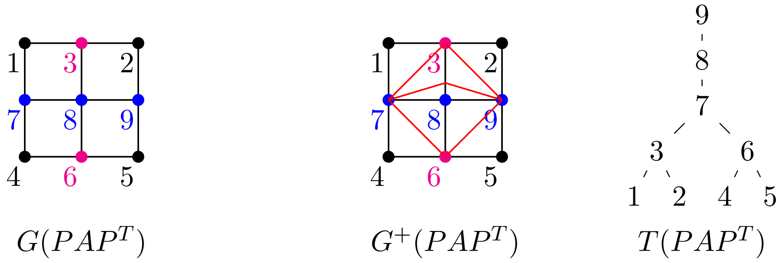

Figure 2: The graphs and the elimination tree of matrix A from equation (2.1) permuted by nested dissection.

We consider now the QR factorization of a matrix A of size $m \times n$, where Q is an $m \times m$ orthogonal matrix and R is an $m \times n$ upper triangular matrix with nonnegative diagonal entries. We consider that the factorization is computed by using Householder transformations. The normal equations $A^T Ax = A^T b$ relate the Cholesky factorization to the QR factorization for full-rank least squares problems. If we ignore numerical cancellations, the graph of R has the same structure as the graph of the Cholesky factor L^T of $A^T A$. The undirected graph of $A^T A$, referred to as the column intersection graph $G_\cap(A)$, has a vertex for each column of A and a clique for each row of A.

3 Multiplication of sparse matrices with random sparsity pattern

In this section we review existing results studying the communication complexity of the multiplication of two sparse matrices $C = A \cdot B$. This is a more difficult problem than studying the communication complexity of the Cholesky factorization of a sparse matrix arising from a model problem. For example, if A and B are block diagonal matrices, then their parallel multiplication can require no communication. Capturing this in a lower bound for matrices with arbitrary sparsity structure is a challenging problem. Hence, existing results consider matrices with particular sparsity structures. We review here results published in [2, 3] which consider matrices with random sparsity pattern, that is matrices whose graphs belong to the class of Erdős - Rényi(n,d) graphs. Given such an $n \times n$ matrix A, referred to as ER(n,d) matrix, each element of A is nonzero with probability d/n. We assume $d \ll n$. The expected number of nonzeros in A is dn. If B is also an ER(n,d) matrix, then the multiplication $C = A \cdot B$ has the following properties: the expected number of multiplications performed is d^2n, the expected number of nonzeros in C is $d^2n(1 - o(1))$.

The general lower bounds on parallel matrix multiplication from [5] can be applied to the case of ER(n,d) matrices and lead to the following bound on volume of communication,

$$\#words \geq \Omega\left(\frac{\#flops}{M^{3/2}} \cdot \frac{M}{P}\right) = \Omega\left(\frac{d^2n}{P\sqrt{M}}\right) \tag{3.1}$$

where P is the number of processors and M is the memory per processor. The results in [3] show that the lower bounds on communication for multiplying two ER(n,d) matrices A and B are

$$\#words \geq \Omega\left(\min\left(\frac{dn}{\sqrt{P}}, \frac{d^2n}{P}\right)\right) = \Omega\left(\frac{dn}{\sqrt{P}}\min\left(1, \frac{d}{\sqrt{P}}\right)\right). \tag{3.2}$$

This bound is obtained by assuming that the sparse matrix multiplication algorithm is sparsity independent, that is the input and output matrices and the computation is distributed over processors independently of the sparsity pattern of the matrices. It also assumes that the input and output matrices are sparse ($d \ll n$) and that the algorithm is load balanced. The bound involves the minimum over two quantities; which bound applies depends on the ratio $d/\sqrt{P}$. This bound improves by a factor of $\sqrt{M} \cdot \max\{1, \sqrt{P}/d\}$ the bound in equation (3.1). Two algorithms introduced in [3] attain this bound. Both algorithms are 3D algorithms, an approach introduced in the context of dense matrix multiplication [1] in which the input and output matrices are distributed over a 3-dimensional grid of processors. The first algorithm is an iterative 3D algorithm that adapts the 3D dense matrix multiplication algorithm from [22] to the sparse case. In the dense case 3D algorithms involve storing multiple copies of the data on processors and thus need extra memory, but in the sparse case there is no extra memory requirement. The second algorithm is a recursive matrix multiplication algorithm which uses ideas from [8]. At each level of the recursion, the multiplication is split into 4 subproblems. The splitting is based on either replicating parts of A and B or redistributing and reducing parts of C, depending on the dimensions of the matrices to be multiplied, such that the communication is minimized.

4 Sparse factorizations

In this section we discuss bounds on communication for direct methods of factorization of sparse matrices.

4.1 Sparse factorizations of matrices whose graphs have small separators

We consider the case of matrices whose graphs have small separators. We consider a matrix A of size $k^s \times k^s$ which is obtained from a finite difference operator on a regular grid of dimension $s \geq 2$ with k^s nodes. The graph of this matrix is defined as a $k \times k \times \ldots \times k$ (s times) mesh, where each node is connected to its neighbours. In this case, it is known that its Cholesky factor contains a dense lower triangular matrix of size $k^{s-1} \times k^{s-1}$ (Lemma 2 in [17]). In addition, the computaton of the Cholesky factorization of this $k^{s-1} \times k^{s-1}$ matrix dominates the computation of the overall Cholesky factorization (Theorem 10 in [17]). In our example matrix from equation (2.3), the matrix formed by the nodes $\{7, 8, 9\}$ is dense lower triangular in the factor L. By using lower bounds on communication required for the computation of the Cholesky factorization of this matrix of dimension $k^{s-1} \times k^{s-1}$, we obtain lower bounds on communication for the overall Cholesky factorization of matrix A. They are given in the following theorem in a more general setting.

Theorem 1 ([15]). *Consider the Cholesky factorization LL^T of an $n \times n$ symmetric matrix A whose undirected graph $G = (V, E)$ has the following property for some l: every set of vertices $W \subset V$ with $n/3 \leq |W| \leq 2n/3$ is adjacent to at least l vertices in $V - W$. A lower bound on communication for computing the Cholesky factorization of A is*

$$\#words \geq \Omega\left(\frac{W}{\sqrt{M}}\right), \qquad \#messages \geq \Omega\left(\frac{W}{M^{3/2}}\right) \tag{4.1}$$

For a sequential algorithm, $W = l^3$ and M is the fast memory size. For a parallel algorithm executed on P processors that is work-balanced, $W = \frac{l^3}{P}$. We assume that the matrix and the L factor are distributed evenly over all the processors and the local memory size used is estimated to be $M = \Theta(nnz(L)/P)$.

For the sequential Cholesky factorization of a $k^s \times k^s$ matrix resulting from a finite difference operator on a regular grid of dimension $s \geq 2$ with k^s nodes, the lower bounds from equation (4.1) apply with $W = k^{3(s-1)}/3$ and M is the fast memory size. In the case of a parallel algorithm executed on P processors in a balanced way, $W = \frac{k^{3(s-1)}}{3P}$ and $M \approx nnz(L)/P$.

These bounds allow us to show that an existing parallel solver for SPD matrices, PSPASES [16], can attain the lower bounds on communication for 2D and 3D regular grids. We explain in the following briefly why. PSPASES reorders the input matrix by using nested dissection and obtains a separator tree (a more compact version of the elimination tree, each node in the separator tree corresponds to a separator obtained during nested dissection). Each node k in the separator tree has a frontal matrix F_k associated with it, which is formed by rows that have their first nonzero in columns of A associated with node k and contribution blocks coming from the children of node k in the separator tree. The numeric factorization in PSPASES is performed during a bottom up traversal of the separator tree. For each frontal matrix F_k, the first rows and columns corresponding to the rows/columns associated with node k are factored and the remaining rows/columns are updated and then transferred to the parent node. At the parent node, the contribution blocks received from the children nodes are added through an extend-add operation. The frontal matrices are distributed over processors by using the subtree to subcube mapping [13] which assigns subsets of processors to nodes in the separator tree. PSPASES uses a bitmask based cyclic distribution such that the communication of contribution blocks between children and parent nodes and the extend-add operation requires point to point communication between pairs of processors which exchange half of their data. With this, the communication performed during the Cholesky factorization of the frontal matrix is greater than the communication required for forming the frontal matrix.

If the Cholesky factorization of every dense multifrontal matrix is computed by using an optimal dense Cholesky factorization [4], the volume of communication, number of messages, and number of flops performed by PSPASES is given in Table 1. An optimal dense Cholesky factorization uses an optimal layout, that is an optimal distribution of the matrix over processors such that the communication is minimized. For more details, the reader can refer to [4]. The results consider an $n \times n$ matrix resulting from a finite difference operator on regular 2D and 3D grids. The analysis assumes the local memory per processor is $M = O(n \log n/P)$ in the 2D case and $M = O(n^{4/3}/P)$ in the 3D case.

Table 1: Lower bounds on communication for 2D and 3D model problems and communication cost of PSPASES solver.

	PSPASES	PSPASES with optimal layout	Lower bound
2D grids			
# flops	$O\left(\frac{n^{3/2}}{P}\right)$	$O\left(\frac{n^{3/2}}{P}\right)$	$\Omega\left(\frac{n^{3/2}}{P}\right)$
# words	$O(\frac{n}{\sqrt{P}})$	$O\left(\frac{n}{\sqrt{P}} \log P\right)$	$\Omega\left(\frac{n}{\sqrt{P \log n}}\right)$
# messages	$O(\sqrt{n})$	$O\left(\sqrt{P} \log^3 P\right)$	$\Omega\left(\frac{\sqrt{P}}{(\log n)^{3/2}}\right)$
3D grids			
# flops	$O\left(\frac{n^2}{P}\right)$	$O\left(\frac{n^2}{P}\right)$	$\Omega\left(\frac{n^2}{P}\right)$
# words	$O(\frac{n^{4/3}}{\sqrt{P}})$	$O\left(\frac{n^{4/3}}{\sqrt{P}} \log P\right)$	$\Omega\left(\frac{n^{4/3}}{\sqrt{P}}\right)$
# messages	$O(n^{2/3})$	$O\left(\sqrt{P} \log^3 P\right)$	$\Omega\left(\sqrt{P}\right)$

Similarly, in [15] it is shown that a sequential multifrontal approach can be derived such that the transfer of data between slow memory and fast memory can be minimized.

In the following, we extend this result to the QR factorization of an $m \times n$ matrix A. The filled column intersection graph $G^+_\cap(A)$ is the graph of the Cholesky factor of A^TA and hence it is also the graph of the R factor. We consider the case when $G_\cap(A)$ belongs to the class of graphs having small separators. We use the lower bounds on communication for computing the Cholesky factor of A^TA to obtain lower bounds on communication for computing the QR factorization of A. This result is presented in the following theorem.

Theorem 2. *Let A be an $m \times n$ matrix such that $G_\cap(A)$ has the following property for some l: every set of vertices $W \subset V$ with $n/3 \leq |W| \leq 2n/3$ is adjacent to at least l vertices in $V - W$. A lower bound on communication for computing the QR factorization of A is*

$$\#words \geq \Omega\left(\frac{W}{\sqrt{M}}\right), \qquad \#messages \geq \Omega\left(\frac{W}{M^{3/2}}\right). \tag{4.2}$$

For a sequential algorithm, $W = l^3$ and M is the fast memory size. For a parallel algorithm executed on P processors that is work-balanced, $W = \frac{l^3}{P}$. We assume that the matrix and the R factor are distributed evenly over all the processors and the local memory size used is estimated to be $M = \Theta(nnz(R)/P)$.

Results in [14] estimate the number of nonzeros in the factors for the graphs considered here. When $G_\cap(A)$ is a structured 2D grid, it is shown in [14] that there exists a column permutation P such that the matrices Q, R, and H in the thin QR factorization of AP satisfy the following bounds: $nnz(R) = O(n \log n)$, $nnz(Q) = O(m\sqrt{n})$, $nnz(H) = O(n \log n + (m-n)\sqrt{n})$.

4.2 Sparse factorizations of matrices whose graphs don't have small separators

We consider the case of matrices whose graphs do not have small separators. In this case, we use a result from [17].

Theorem 3 (Theorems 13 and 14 from [17])**.** *For all $\epsilon > 0$ there is a constant $c(\epsilon)$ such that almost all n-vertex undirected graphs with at least $c(\epsilon)n$ edges have a fill-in clique of at least $(1-\epsilon)n$ vertices for any ordering. These graphs have a fill-in of $(1-\epsilon)^2 n^2/2 - O(n)$ and a multiplication count of $(1-\epsilon)^3 n^3/6 - O(n^2)$ for any ordering.*

For the Cholesky factorization of sparse symmetric positive definite (SPD) matrices, the clique identified in these n-vertex graphs leads to a dense submatrix in the factor L of Cholesky factorization. We use this dense submatrix to compute lower bounds on communication for sequential and parallel Cholesky factorizations in Theorem 4. More generally, these results also apply if Gaussian elimination is used with some form of symmetric pivoting which does not destroy the symmetry of the input matrix.

Theorem 4. *Consider the Cholesky factorization of an SPD $n \times n$ matrix A whose graph satisfies the conditions of Theorem 3, that is the filled graph $G^+(A)$ has a clique of $(1-\epsilon)n$ vertices for any ordering. A lower bound on communication for computing the Cholesky factorization of A is:*

$$\#words \geq \Omega\left(\frac{(1-\epsilon)^3 n^3}{\sqrt{M}}\right), \qquad \#messages \geq \Omega\left(\frac{(1-\epsilon)^3 n^3}{M^{3/2}}\right), \tag{4.3}$$

where M is the fast memory size and we omit some lower order terms. For the parallel factorizations on P processors, we consider that the algorithm is memory- and work-balanced and that $M = O((1-\epsilon)^2 n^2)/P)$ is the local memory used per processor. At least one processor communicates:

$$\#words \geq \Omega\left(\sqrt{P}M\right), \qquad \#messages \geq \Omega\left(\sqrt{P}\right). \tag{4.4}$$

Proof. The undirected graph of A^TA, which is an SPD matrix, satisfies the properties in Theorem 3 which identify a dense submatrix in L^T. The communication necessary for computing this dense submatrix in L^T represents a lower bound on communication for computing the entire Cholesky factorization. □

We can use the same reasoning as in section 4.1 for deriving lower bounds on communication for the QR factorization. Since the graph of R is the same as the graph of the Cholesky factor of A^TA, if the graph of A^TA satisfies the conditions in Theorem 3, then there is a dense triangular submatrix in R of dimension $(1-\epsilon)n \times (1-\epsilon)n$. The communication involved in factoring this submatrix gives a lower bound on the communication of the entire matrix.

We conjecture that sparse sequential and parallel implementations of sequential recursive Cholesky [11], parallel Cholesky as implemented in ScaLAPACK, CAQR [**?**] with an optimal layout attain the lower bounds on communication in Theorem 4.

5 Conclusions

In this report we have summarized several results on the communication complexity of sparse matrix operations. We first reviewed results on lower bounds on communication for the multiplication of two matrices with random sparsity structure. We then discussed lower bounds on communication for the Cholesky and QR factorizations of matrices with small separators and algorithms that attain these bounds. Finally we introduced lower bounds on communication for matrices whose adjancency graphs do not have small separators.

References

[1] R. C. Agarwal, S. M. Balle, F. G. Gustavson, M. Joshi, and P. Palkar. A three-dimensional approach to parallel matrix multiplication. *IBM Journal of Research and Development*, 39(5):575 –582, sep. 1995.

[2] A. Azad, G. Ballard, A. Buluc, J. Demmel, L. Grigori, O. Schwartz, S. Toledo, and S. Williams. Exploiting multiple levels of parallelism in sparse matrix-matrix multiplication. *SIAM Journal on Scientific Computing*, in press, 2016.

[3] G. Ballard, A. Buluc, J. Demmel, L. Grigori, O. Schwartz, and S. Toledo. Communication optimal parallel multiplication of sparse random matrices. In *Proceedings of ACM SPAA, Symposium on Parallelism in Algorithms and Architectures*, 2013.

[4] G. Ballard, J. Demmel, O. Holtz, and O. Schwartz. Communication-optimal parallel and sequential Cholesky decomposition. *SIAM Journal on Scientific Computing*, 32(6):3495–3523, December 2010.

[5] G. Ballard, J. Demmel, O. Holtz, and O. Schwartz. Minimizing communication in linear algebra. *SIAM J. Matrix Anal. Appl.*, 2011.

[6] A. Buluç and J. R. Gilbert. Parallel sparse matrix-matrix multiplication and indexing: Implementation and experiments. *SIAM Journal of Scientific Computing (SISC)*, 34(4):170 – 191, 2012.

[7] E. Carson, J. Demmel, L. Grigori, N. Knight, P. Koanantakool, O. Schwartz, and H. V. Simhadri. Write-avoiding algorithms. In *Proceedings of IEEE International Parallel and Distributed Processing Symposium IPDPS*, pages 648–658, 2016.

[8] J. Demmel, D. Eliahu, A. Fox, S. Kamil, B. Lipshitz, O. Schwartz, and O. Spillinger. Communication-optimal parallel recursive rectangular matrix multiplication. In *International Parallel & Distributed Processing Symposium (IPDPS)*. IEEE, 2013.

[9] J. Demmel, L. Grigori, M. Hoemmen, and J. Langou. Communication-optimal parallel and sequential QR and LU factorizations. Technical Report UCB/EECS-2008-89, UC Berkeley, 2008. LAPACK Working Note 204.

[10] I. S. Duff. Full matrix techniques in sparse Gaussian elimination. In Springer-Verlag, editor, *Lecture Notes in Mathematics (912)*, pages 71–84, 1982.

[11] M. Frigo, C. E. Leiserson, H. Prokop, and S. Ramachandran. Cache-oblivious algorithms. In IEEE Computer Society, editor, *FOCS'99: Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, page 285, 1999.

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

[13] A. George, J. W.-H. Liu, and E. G. Ng. Communication results for parallel sparse Cholesky factorization on a hypercube. *Parallel Computing*, 10(3):287–298, 1989.

[14] J. R. Gilbert, E. G. Ng, and B. W. Peyton. Separators and structure prediction in sparse orthogonal factorization. *Linear Algebra and its Applications*, 262, 1997.

[15] L. Grigori, P.-Y. David, J. Demmel, and S. Peyronnet. Brief announcement: Lower bounds on communication for direct methods in sparse linear algebra. *Proceedings of ACM SPAA*, 2010.

[16] A. Gupta, G. Karypis, and V. Kumar. Highly scalable parallel algorithms for sparse matrix factorization. *IEEE Transactions on Parallel and Distributed Systems*, 8(5), 1995.

[17] R. J. Lipton, D. J. Rose, and R. E. Tarjan. Generalized nested dissection. *SIAM Journal on Numerical Analysis*, 16:346–358, 1979.

[18] S. Parter. The use of linear graphs in Gaussian elimination. *SIAM Review*, pages 364–369, 1961.

[19] D. J. Rose. Triangulated graphs and the elimination process. *Journal of Mathematical Analysis and Applications*, pages 597–609, 1970.

[20] D. J. Rose and R. E. Tarjan. Algorithmic aspects of vertex elimination on directed graphs. *SIAM J. Appl. Math.*, 34(1):176–197, 1978.

[21] R. Schreiber. A new implementation of sparse Gaussian elimination. *ACM Trans. Math. Software*, 8:256–276, 1982.

[22] E. Solomonik, A. Bhatele, and J. Demmel. Improving communication performance in dense linear algebra via topology aware collectives. In *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, page 77. ACM, 2011.

[23] E. Solomonik and J. Demmel. Communication-optimal parallel 2.5D matrix multiplication and LU factorization algorithms. In *Euro-Par'11: Proceedings of the 17th International European Conference on Parallel and Distributed Computing*. Springer, 2011.

[24] R. A. van de Geijn and J. Watts. SUMMA: Scalable universal matrix multiplication algorithm. *Concurrency - Practice and Experience*, 9(4):255–274, 1997.