PSPASES: An Efficient and Scalable Parallel Sparse Direct Solver *

Mahesh Joshi George Karypis Vipin Kumar
Department of Computer Science
University of Minnesota
Minneapolis, MN 55455

Anshul Gupta Fred Gustavson
Mathematical Sciences Department
IBM T.J.Watson Research Center
Yorktown Heights, NY 10598

Abstract

Many problems in engineering and scientific domains require solving large sparse systems of linear equations, as a computationally intensive step towards the final solution. It has long been a challenge to develop efficient parallel formulations of sparse direct solvers due to several different complex steps involved in the process. In this paper, we describe PSPASES, one of the first efficient, portable, and robust scalable parallel solvers for sparse symmetric positive definite linear systems that we have developed. We discuss the algorithmic and implementation issues involved in its development; and present performance and scalability results on Cray T3E and SGI Origin 2000. PSPASES could solve the largest sparse system (1 million equations) ever solved by a direct method, with the highest performance (51 GFLOPS for Cholesky factorization) ever reported.

1 Introduction

Solving large sparse systems of linear equations is at the heart of many engineering and scientific computing applications. There are two methods to solve these systems - direct and iterative. Direct methods are preferred for many applications because of various properties of the method and the nature of the application. A wide class of sparse linear systems arising in practice have a symmetric positive definite (SPD) coefficient matrix. The problem is to compute the solution to the system $Ax = b$, where A is a sparse and SPD matrix. Such a system is commonly solved using Cholesky factorization. A direct method of solution consists of four consecutive phases viz. ordering, symbolic factorization, numerical factorization and solution of triangular systems. During the ordering phase, a permutation matrix P is computed so that the matrix PAP^T will incur a minimal fill during the factorization phase. During the symbolic factorization phase, the non-zero structure of the triangular Cholesky factor L is determined. The symbolic factorization phase exists in order to increase the performance of the numerical factorization phase. The necessary operations to compute the values in L that satisfy $PAP^T = LL^T$, are performed during the phase of numerical factorization. Finally, the solution to $Ax = b$ is computed by solving two triangular systems viz. $Ly = b'$ followed by

*This work was supported by NSF CCR-9423082, by Army Research Office contract DA/DAAG55-98-1-0441, by Army High Performance Computing Research Center cooperative agreement number DAAH04-95-2-0003/contract number DAAH04-95-C-0008, by the IBM Partnership Award, and by the IBM SUR equipment grant. Access to computing facilities was provided by AHPCRC, Minnesota Supercomputer Institute. Related papers are available via WWW at URL: http://www.cs.umn.edu/~kumar.

$L^T x' = y$, where $b' = Pb$ and $x' = Px$. Solving the former system is called forward elimination and the latter process of solution is called backward substitution. The final solution, x, is obtained using $x = P^T x'$.

In this paper, we describe one of the first scalable and high performance parallel direct solvers for sparse linear systems involving SPD matrices. At the heart of this solver is a highly parallel algorithm based on multifrontal Cholesky factorization we recently developed [1]. This algorithm is able to achieve high computational rates (of over 50 GFLOPS on a 256 processor Cray T3E) and it successfully parallelizes the most computationally expensive phase of the sparse solver. Fill reducing ordering is obtained using a parallel formulation of the multilevel nested dissection algorithm [5] that has been found to be effective in producing orderings that are suited for parallel factorization. For symbolic factorization and solution of triangular systems, we have developed parallel algorithms that utilize the same data distribution as used by the numerical factorization algorithm. Both algorithms are able to effectively parallelize the corresponding phases. In particular, our parallel forward elimination and parallel backward substitution algorithms are scalable and achieve high computational rates [3].

We have integrated all these algorithms into a software library, PSPASES (Parallel SPArse Symmetric dirEct Solver). It uses the MPI library for communication, making it portable to a wide range of parallel computers. Furthermore, high computational rates are achieved using serial BLAS routines to perform the computations at each processor. Various easy-to-use functional interfaces, and internal memory management are among other features of PSPASES.

In the following, first we briefly describe the parallel algorithms for each of the phases, and their theoretical scalability behavior. Then, we describe the functional interfaces and features of the PSPASES library. Finally, we demonstrate its performance and scalability for a wide range of matrices on Cray T3E and SGI Origin 2000 machines.

2 Algorithms

PSPASES implements scalable parallel algorithms in each of the phases. In essence, the algorithms are driven by the elimination tree structure of A [9]. The ordering phase computes an ordering with the aims of reducing the fill-ins and generating a balanced elimination tree. Symbolic factorization phase traverses the elimination tree in a bottom-up fashion to compute the non-zero structure of L. Numerical factorization phase is based on a highly parallel multifrontal algorithm. This phase determines the internal data distribution of A and L matrices. The triangular solve phase also uses multifrontal algorithms. Following subsections give more details of each of the phases.

2.1 Parallel Ordering

We use our multilevel nested dissection based algorithm for computing fill-reducing ordering.

The matrix A is converted to its equivalent graph, such that each row i is mapped to a vertex i of the graph and each nonzero $a_{ij} (i \neq j)$ is mapped to an edge between vertices i and j. A parallel graph partitioning algorithm [5] is used to compute a high quality p-way partition, where p is the number of processors. The graph is then redistributed according to this partition. This pre-partitioning phase helps to achieve high performance for computing the ordering, which is done by computing the $\log p$ levels of the elimination tree concurrently. At each level, multilevel paradigm is used to partition the vertices of the graph. The set of nodes lying at the partition boundary forms a *separator*. Separators at all levels are stored. When the graph is separated into p parts, it is redistributed among the processors such that each processor receives a single subgraph, which it orders using the multiple minimum degree (MMD) algorithm. Finally, a global numbering is imposed on all nodes such that nodes in each separator and each subgraph are numbered consecutively, respecting the MMD ordering.

Details of the multilevel paradigm can be found in [5]. Briefly, it involves gradual coarsening of a graph to a few hundred vertices, then partitioning this smaller graph, and finally, projecting the partitions back to the original graph by gradually refining them. Since the finer graph has more degrees of freedom, such refinements improve the quality of the partition.

When a matrix is reordered according to the ordering generated by above algorithm, it has a

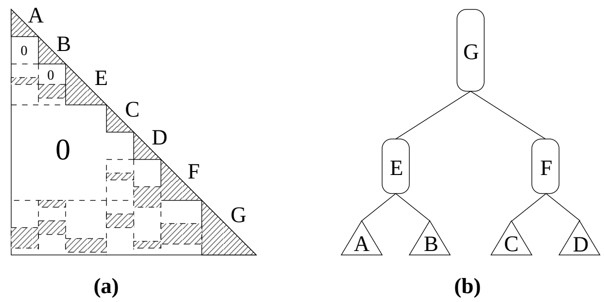

FIG. 1. *Ordering a system for 4 processors. (a) Structure of a reordered matrix, (b) Separator tree corresponding to it.*

format similar to that shown in Figure 1(a). The separator tree corresponding to the reordered graph has a structure of the form shown in Figure 1(b).

2.2 Parallel Numerical Factorization

We will first describe the parallel numerical factorization phase, because algorithm used here decides the data distribution of L, and hence drives the symbolic factorization algorithm. We use our highly scalable algorithm [1] that is based on the multifrontal algorithm [8].

Given a sparse matrix and the associated elimination tree, the multifrontal algorithm can be recursively formulated as follows. Consider an $N \times N$ matrix A. The algorithm performs a postorder traversal of the elimination tree associated with A. There is a frontal matrix F^k and an update matrix U^k associated with any node k. The row and column indices of F^k correspond to the indices of row and column k of L, the lower triangular Cholesky factor, in increasing order. In the beginning, F^k is initialized to an $(s+1) \times (s+1)$ matrix, where $s+1$ is the number of non-zeros in the lower triangular part of column k of A. The first row and column of this initial F^k is simply the upper triangular part of row k and the lower triangular part of column k of A. The remainder of F^k is initialized to all zeros.

After the algorithm has traversed all the subtrees rooted at a node k, it ends up with a $(t+1) \times (t+1)$ frontal matrix F^k, where t is the number of non-zeros in the strictly lower triangular part of column k in L. The row and column indices of the final assembled F^k correspond to $t+1$ (possibly) noncontiguous indices of row and column k of L in increasing order. If k is a leaf in the elimination tree of A, then the final F^k is the same as the initial F^k. Otherwise, the final F^k for eliminating node k is obtained by merging the initial F^k with the update matrices obtained from all the subtrees rooted at k via an extend-add operation. The extend-add is an associative and commutative operator on two update matrices such the index set of the result is the union of the index sets of the original update matrices. Each entry in the original update matrices is mapped onto some location in the accumulated matrix. If entries from both matrices overlap on a location, they are added. Empty entries are assigned a value of zero. After F^k has been assembled, a single step of the standard dense Cholesky factorization is performed with node k as the pivot. At the end of the elimination step, the column with index k is removed from F^k and forms the column k of L. The remaining $t \times t$ matrix is called the update matrix U^k and is passed on to the parent of k in the elimination tree.

A collection of consecutive nodes in the elimination tree, each with only one child, is called a *supernode*. Nodes in each supernode are collapsed together to form a *supernodal elimination tree*, which corresponds to the separator tree [Figure 1(b)] in the top $\log p$ levels.

The serial multifrontal algorithm can be extended to operate on this supernodal tree by extending the single node operations performed while forming and factoring the frontal matrix. The frontal matrix corresponding to a supernode with l nodes is obtained by merging the frontal matrices of the individual nodes, and the first l columns of this frontal matrix are factored during the factorization of this supernode.

In our parallel formulation of the multifrontal algorithm, we assume that the supernodal tree

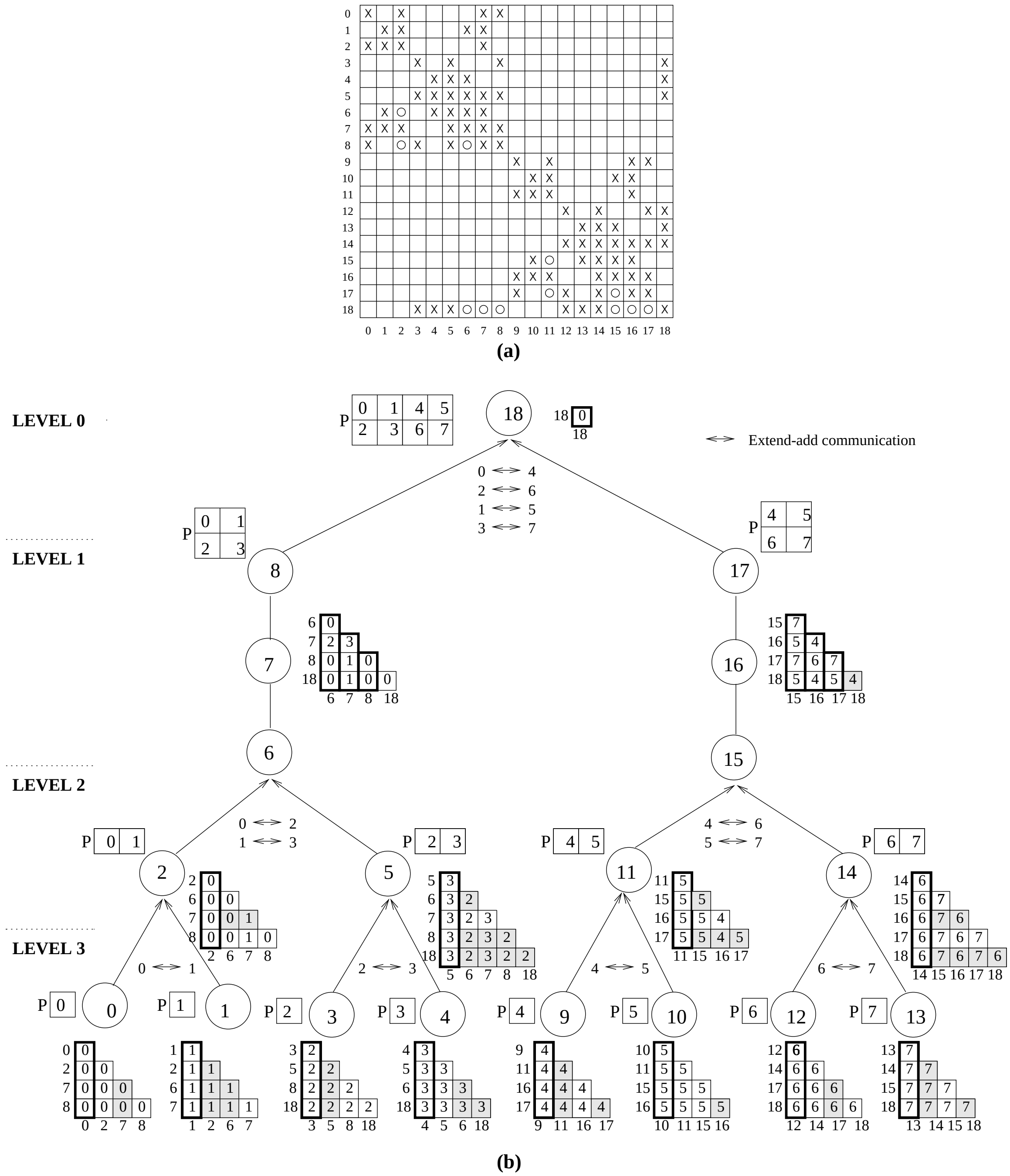

FIG. 2. **(a).** *An example symmetric sparse matrix. The non-zeros of A are shown with symbol "$\times$" in the upper triangular part and non-zeros of L are shown in the lower triangular part with fill-ins denoted by the symbol "$\circ$".* **(b).** *The process of parallel multifrontal factorization using 8 processors. At each supernode, the factored frontal matrix, consisting of columns of L (thick columns) and update matrix (remaining columns), is shown.*

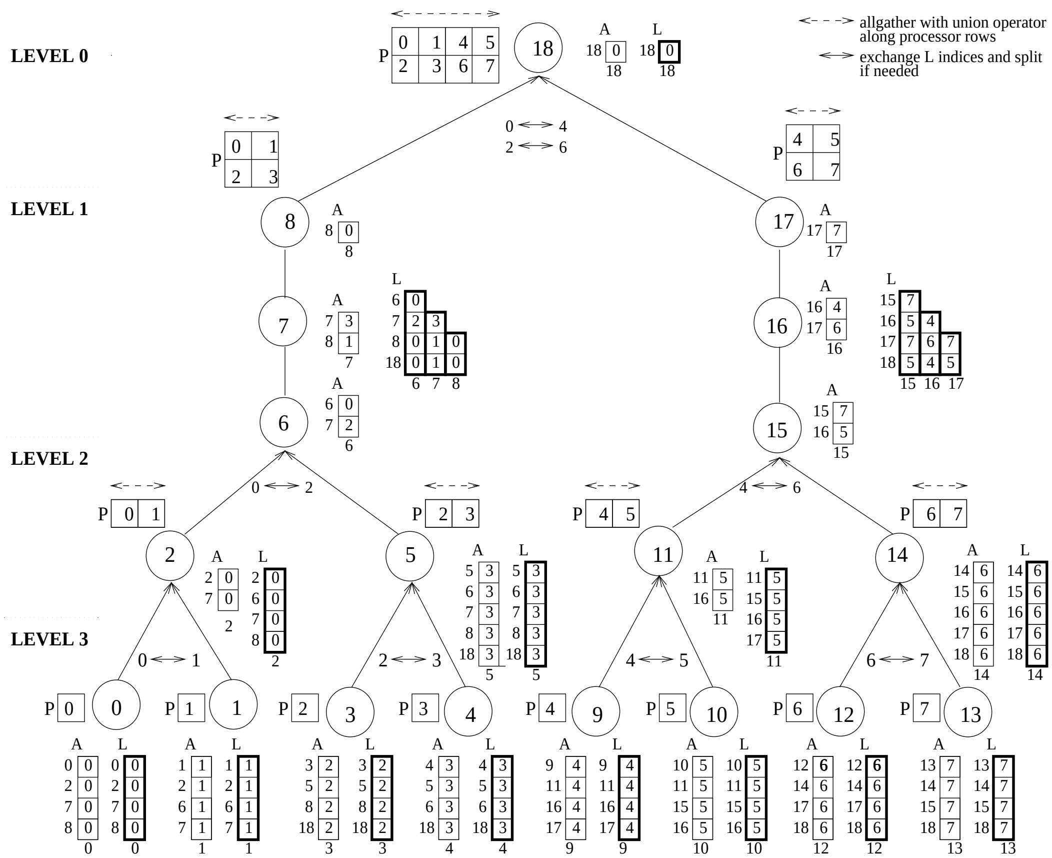

FIG. 3. *Parallel Symbolic Factorization*

is binary in the top $\log p$ levels [1]. The portions of this binary supernodal tree are assigned to processors using a subtree-to-subcube strategy illustrated in Figure 2(b), where eight processors are used to factor the example matrix of Figure 2(a). The subcubes of processors working on various subtrees are shown in the form of a logical mesh labeled with P. The frontal matrix of each supernode is distributed among this logical mesh using a bitmask based block-cyclic scheme [1]. Figure 2(b) shows such a distribution for unit blocksize. This distribution ensures that the extend-add operations required by the multifrontal algorithm can be performed in parallel with each processor exchanging roughly half of its data *only* with its partner from the other subcube. Figure 2(b) shows the parallel extend-add process by showing the pairs of processors that communicate with each other. Each processor sends out the shaded portions of the update matrix to its partner. The parallel factor operation at each supernode is a pipelined implementation of the dense column Cholesky factorization algorithm.

2.3 Parallel Symbolic Factorization

During the symbolic factorization phase the non-zero structure of the factor matrix L is determined. The serial algorithm to generate the structure of L performs a postorder traversal of the elimination tree. At each node k, the L indices of all its children node (excluding the children nodes themselves) and the A indices of k are merged together to form the L indices of node k.

This algorithm can be effectively parallelized using the same subtree-to-subcube mapping used by the numerical factorization algorithm. The basic structure of the algorithm is illustrated in

[1]The separator tree obtained from the recursive nested dissection parallel ordering algorithm used in our solver yields such a binary tree (Fig. 1(b).

Figure 3. Initially, matrix A is distributed such that the columns of A of each supernode are distributed using the same bitmask based block-cyclic distribution required by the numerical factorization phase [2]. Now, the non-zero structure of L is determined in a bottom-up fashion. First the non-zero structure of the leaf nodes is determined and it is sent upwards in the tree, to the processors that store the next level supernode. These processors determine the non-zero structure of their supernode and merge it with the non-zero structure received from their children nodes. For example, consider the computation involved in determining the structure of L for the supernode consisting of the nodes {6,7,8} which are distributed among a 2×2 processor grid. First, the processors determine the non-zero structure of L by performing a union along the rows of the grid to collect the distinct row indices of the columns of A corresponding to the nodes {6,7,8}. The result is {6,8} and {7} at the first and second row of processors, respectively. Now the non-zero structure of the children nodes are received (excluding the nodes themselves). Since these non-zero structures are stored in the two 1×2 sub-grids of the 2×2 grid, this information is sent using a similar communication pattern described in Section 2.2, however only the processors in the first column of the subgrids need to communicate. In particular, processor 0 splits the list {6,7,8} into two parts {6,8} and {7}, retains {6,8} and sends {7} to processor 2. Similarly processor 2 splits the list {6,7,8,18} into two parts {6,8,18} and {7}, retains {7} and sends {6,8,18} to processor 0. Now processors 0 and 2 merge these lists. In particular, processor 0 merges {6,8}, {6,8} and {6,8,18} and processor 2 merges {7}, {7} and {7}.

2.4 Parallel Triangular Solve

During the phase of solving triangular systems, a forward elimination $Ly = b'$, where $b' = Pb$, is performed followed by a backward substitution $L^T x' = y$ to determine the solution $x = P^T x'$. Our parallel algorithms for this phase are guided by the supernodal elimination tree. They use the same subtree-to-subcube mapping and the same two-dimensional distribution of the factor matrix L as used in the numerical factorization.

Figure 4(a) illustrates the parallel formulation of the forward elimination process. The right hand side vector b', is distributed to the processors that own the corresponding diagonal blocks of the L matrix as shown in the shaded blocks in Figure 4(a). The computation proceeds in a bottom-up fashion. Initially, for each leaf node k, the solution y_k is computed and is used to form the update vector $\{l_{ik}y_k\}$ (denoted by "U" in Figure 4(a)). The elements of this update vector need to be subtracted from the corresponding elements of b', in particular $l_{ik}y_k$ will need to be subtracted from b'_i. However, our algorithm uses the structure of the supernodal tree to accumulate these updates upwards in the tree and subtract them only when the appropriate node is being processed. For example consider the computation involved while processing the supernode {6,7,8}. First the algorithm merges the update vectors from the children supernodes to obtain the combined update vector for indices {6,7,8,18}. Note that the updates to the same b' entries are added up. Then it performs forward elimination to compute y_6, y_7 and y_8. This computation is done using a two dimensional pipelined dense forward elimination algorithm. At the end of the computation, the update vector on processor 0 contains the updates for for b'_{18} due to y_6, y_7 and y_8 as well as the updates received from supernode {5}. In general, at the end of the computation at each supernode, the accumulated update vector resides on the column of processors that store the last column of the L matrix of that supernode. This update vector needs to be sent to the processors that store the first column of the L matrix of the parent supernode. Because of the bitmask based block-cyclic distribution, this can be done by using at most two communication steps [3].

The details of the two-dimensional pipelined dense forward elimination algorithm are illustrated in Figure 4(b) for a hypothetical supernode. The solutions are computed by the processors owning diagonal elements of L matrix and flow down along a column. The accumulated updates flow along the row starting from the first column and ending at the last column of the supernode. The processing is pipelined in the shaded regions and in other regions the updates are accumulated using a reduction operation along the direction of the flow of update vector.

[2]This distribution is performed after computing ordering.

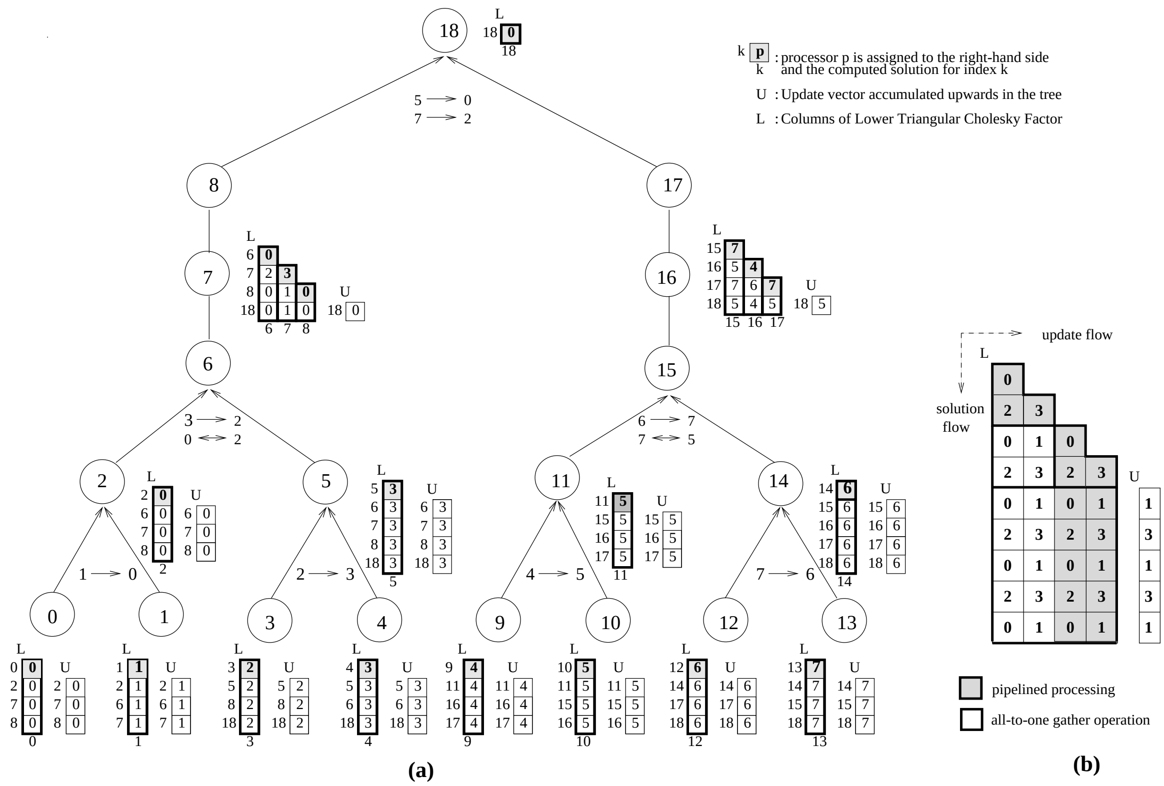

FIG. 4. *Parallel Triangular Solve.* **(a).** *Entire process of parallel forward elimination for the example matrix.* **(b).** *Processing within a hypothetical supernodal matrix for forward elimination.*

The algorithm for parallel backward substitution is similar, except for two differences. First, the computation proceeds from the top supernode of the tree down to the leaf. Second, the computed solution that gets communicated across the levels of the supernodal tree instead of accumulated updates and this is achieved with at most one communication per processor. Refer to [3] for details.

2.5 Parallel Runtime and Scalability Analysis

Analyzing a general sparse solver is a difficult problem. We present the analysis for two wide classes of sparse systems arising out of two-dimensional and three-dimensional constant node-degree graphs. We refer to these problems as 2-D and 3-D problems, respectively.

Let a problem of dimension N be solved using p processors. Define l as a level in the supernodal tree with $l = 0$ for the topmost level. Thus, the number of processors assigned to a level l supernode is $p/2^l$. With a nested dissection based ordering scheme, it has been shown that the average size of a frontal matrix at level l is $\theta(N/2^{2l})$ for 2-D problems, and $\theta(N^{4/3}/2^{4l})$ for 3-D problems. Also, the number of nodes in a level-l supernode is on an average $\theta(\sqrt{N/2^l})$ and $\theta((N/2^l)^{2/3})$ for 2-D and 3-D problems, respectively. The number of non-zeros in L is $N \log N$ for 2-D problems, and $N^{4/3}$ for 3-D problems. Using this information, the parallel algorithms presented earlier in this section have been analyzed for their parallel runtime. For details, refer to [2, 3].

Table 1 shows the serial runtime complexity, parallel overhead, and isoefficiency functions [7] for all four phases. It can be seen that all the phases are scalable to varying degrees. Since the overall time complexity is dominated by the numerical factorization phase, the isoefficiency function of the entire parallel solver is determined by the numerical factorization phase, and it is $O(p^{1.5})$ for both 2-D and 3-D problems. As discussed in [7], the isoefficiency function of the dense Cholesky factorization algorithm is also $O(p^{1.5})$. Thus our sparse direct solver is as scalable as the dense factorization algorithm.

	2-D constant node-degree graph			3-D constant node-degree graph		
Phase	T_s	T_o	IsoEff	T_s	T_o	IsoEff
Order	$O(N\log N)$	$O(p^2\log p)$	$O(p^2\log p)$	$O(N\log N)$	$O(p^2\log p)$	$O(p^2\log p)$
SymFact	$O(N\log N)$	$O(\sqrt{Np}\log p)$	$O(p\log p)$	$O(N^{4/3})$	$O(N^{2/3}\sqrt{p})$	$O(p)$
NumFact	$O(N^{3/2})$	$O(N\sqrt{p})$	$O(p\sqrt{p})$	$O(N^2)$	$O(N^{4/3}\sqrt{p})$	$O(p\sqrt{p})$
TriSolve	$O(N\log N)$	$O(\sqrt{N}p)$	$O(p^2/\log p)$	$O(N^{4/3})$	$O(N^{2/3}p)$	$O(p^2)$

TABLE 1

Complexity Analysis of various phases of our parallel solver for N-vertex constant node-degree graphs. Notations: T_s = Serial Runtime Complexity, T_o = Parallel Overhead $= pT_p - T_s$, where T_p is the Parallel Runtime. IsoEff is the isoefficiency function indicating the rate at which the problem size should increase with the number of processors to achieve same efficiency.

3 Library Implementation and Experimental Results

We have implemented all the parallel algorithms presented in previous section, and integrated them into a library called PSPASES [4]. This library uses MPI call interface for communication, and BLAS interface for computation within a processor. This makes PSPASES portable to a wide range of parallel computers. Furthermore, using BLAS, it is able to achieve high computational performance especially on the platforms with vendor-tuned BLAS libraries.

We have incorporated some more features into PSPASES to make it easy-to-use. The library provides simple functional interfaces, and allows user to call its functions from both C and Fortran 90 programs. It uses the concept of a communicator which relieves the user from managing the memory required by various internal data structures. The communicator also allows to solve multiple systems with same sparsity structure, or same system for multiple sets of right hand side matrix, B. PSPASES also provides the user with useful statistical information such as the load imbalance of the supernodal elimination tree, number of fill-ins (both of which can be used to judge the quality of ordering), operation count for factor operation, and estimation of memory for internal data structures.

We have tested PSPASES on a wide range of parallel machines, for a wide range of sparse matrices. Tables 3 through 12 give experimental results obtained on a 64-processor SGI Origin 2000 and a 256-processor Cray T3E-1200, respectively. The Origin 2000 machine had 195MHz MIPS R10000 processors organized into 12 modules. The total memory available was 48 GB among 96 processors, out of which we used 20 GB for 64 processors. The T3E-1200 machine had 600MHz Alpha processors connected via a 3-D torus network capable of 480 MB/sec peak bandwidth in each direction. Each processor had 512MB memory. We used native versions of both the MPI and BLAS libraries. The notations used in presenting the results are described in Table 2. We give numbers for the numerical factorization and triangular solve phases only. The parallel ordering algorithm we use is incorporated into a separate library called ParMetis [6], and PSPASES library functions just call the ParMetis library functions. ParMetis performance numbers are reported elsewhere [6]. The symbolic factorization took relatively very small time, and hence, we do not report those numbers here.

The results are divided into tables for small matrices with $N <$ 100,000 and large matrices for larger values of N. We also divide the results based on serial and parallel ordering. Under normal circumstances, ParMetis gives different orderings for different number of processors, for the same A matrix. We wanted to see the true scalability of the algorithms by using a fixed ordering, so we used the serial ordering option with a little modification to the ParMETIS code in order to get get the same ordering as that obtained on $np = 256$ processors, on all the values of np. These results are reported in the tables 3, 4 for small and large systems on Origin 2000; and in tables 7, 8 and 9 for small and large systems on T3E-1200. The slight increase in the values of number of non-zeros in L, with number of processors is due to the fact that the parallel part of the code (in top *log p* levels) converts a chain of nodes into a single fat supernode, whereas the code which implements the algorithms on leaf subtrees is more judicious in splitting the chain into multiple supernodes. The results with parallel ordering are shown in tables 5 and 6 for Origin 2000 machine; and in tables 10, 11, and 12 for T3E-1200. The number of nonzeros in L is different for all values of np because of the nature of parallel ordering algorithms used. The results for all the phases, including ordering and symbolic factorization, are given in tables 13 and 14 for Origin 2000 when parallel ordering is used. The scalability in the total runtime (PSPtime) is evident of the fact the entire solver taken together is scalable. The increase in ordering time on 64 processors for some matrices (especially small matrices) was looked into in detail, and we found that the MPI library that we were using was not giving proper performance for collective communication operations. Similar behavior was observed for symbolic phase and the redistribution of B and X matrices (before and after triangular solve).

It can be seen that numerical factorization phase demonstrates high performance and scalability for all the matrices. Triangular solve phase also yields a very scalable and high performance in almost all cases, except for some cases involving small matrices being solved on large number of processors, when the amount of computation to communication ratio becomes relatively very small.

The performance is also seen to be varying with the tree imbalance factor. The lower the number in *Imbal* column, more balanced is the tree, as far as the computations in different subtrees are concerned. The *Imbal* factor is computed by taking into account the computational load imbalance in each of the subtrees, and assuming that the subtree with higher computation forces the processors of other subtree to idle. It is computed recursively in a bottom-up fashion. For example, in Figure 2(b), the computations needed to factorize the subtree rooted at supernode $\{2\}$ is equal to the maximum of the computations taken by the subtrees rooted at nodes 0 and 1, plus the time to compute the supernodal matrix. Going one level up, the computations needed to factorize the subtree rooted at the supernode $\{6,7,8\}$ is equal to the computations required factorize that supernode plus the maximum of computations required for trees rooted at supernodes $\{2\}$ and $\{5\}$. The process continues in this manner until the root of the tree is reached. For balanced trees, performance numbers are high because the idling between corresponding groups of communicating processors is lower for such trees. Thus, better ordering which yields balanced trees is desirable for high performance.

It should be noted that along with the balanced trees, the blocksize of distributing supernodal matrices plays an important role in determining the performance of the solver. High performance of the pipelined operations in the supernodal matrices is achieved for optimal blocksize, which is determined by the ratio of computation rate to the communication rate of the underlying parallel system. Also, since most of today's parallel computers have better communication performance for larger data being communicated, blocksize optimal for numerical factorization need not be optimal for triangular solve phase, which has relatively small data communication per computational block. For the results presented, blocks of 1024 or 4096 double precision numbers worked best for numerical factorization phase.

The highest performance of 51 GFLOPS was obtained for numerical factorization for the 1-million equation system CUBE100, on 256 processors of Cray T3E-1200. To our knowledge, this is the largest sparse system ever solved by a direct solution method, with the highest performance ever reported.

We have made PSPASES library and more experimental results available via WWW at URL: **http://www.cs.umn.edu/~mjoshi/pspases**.

References

[1] Anshul Gupta, George Karypis and Vipin Kumar, *Highly Scalable Parallel Algorithms for Sparse Matrix Factorization*, IEEE Transactions on Parallel and Distributed Systems, vol. 8, no. 5, pp. 502-520, 1997.

[2] Anshul Gupta, *Analysis and Design of Scalable Parallel Algorithms for Scientific Computing*, Ph.D. Thesis, Department of Computer Science, University of Minnesota, Minneapolis, 1995.

[3] Mahesh Joshi, Anshul Gupta, George Karypis and Vipin Kumar, *A High Performance Two Dimensional Scalable Parallel Algorithm for Solving Sparse Triangular Systems*, Proc. of 4th Int. Conf. on High Performance Computing (HiPC'97), Bangalore, India, 1997.

N	:	Dimension of A.
NNZ_LowerA	:	Number of non-zeros in Lower triangular part of A (including diagonal).
np	:	Number of Processors.
NNZ_L	:	Number of non-zeros in L.
Imbal	:	Computational Load Imbalance due to supernodal tree structure.
FAC_OPC	:	Operation count for Numerical Factor phase.
Ntime	:	Numerical Factor Time (in seconds).
Nperf	:	Numerical Factor Performance (in MFLOPS).
Ttime	:	Forward Solve Time + Backward Solve Time for nrhs = 1.

TABLE 2

Notations used in the presented results

[4] Mahesh Joshi, George Karypis, Vipin Kumar, Anshul Gupta, and Fred Gustavson, *PSPASES: Scalable Parallel Direct Solver Library for Sparse Symmetric Positive Definite Linear Systems (Version 1.0), User's Manual*, Available via http://www.cs.umn.edu/~mjoshi/pspases.

[5] George Karypis and Vipin Kumar, *Parallel Algorithm for Multilevel Graph Partitioning and Sparse Matrix Ordering*, Journal of Parallel and Distributed Computing, vol. 48, pp. 71-95, 1998.

[6] George Karypis, Kirk Schlogel, and Vipin Kumar *ParMetis: Parallel Graph Partitioning and Sparse Matrix Ordering Library, (Version 1.0), User's Manual*, Available via http://www.cs.umn.edu/~metis.

[7] Vipin Kumar, Ananth Grama, Anshul Gupta and George Karypis, *Introduction to Parallel Computing: Design and Analysis of Algorithms* Benjamin/Cummings, Redwood City, CA, 1994.

[8] Joseph W. H. Liu, *The Multifrontal Method for Sparse Matrix Solution: Theory and Practice*, SIAM Review, vol. 34, no.1, pp. 82-109, March 1992.

[9] Joseph W. H. Liu, *The Role of Elimination Trees in Sparse Factorization*, SIAM Journal of Matrix Analysis and Applications, vol. 11, pp. 134-172, 1990.

np	NNZ_L	Imbal	FAC_OPC	Ntime	Nperf	Ttime
matrix : S3DKQ4M2, N: 90449, NNZ_LowerA: 2.4600E+06						
2	1.74e+07	1.00	6.81e+09	18.756	363.07	1.163
4	1.74e+07	1.02	6.81e+09	9.364	727.21	0.689
8	1.74e+07	1.17	6.81e+09	6.138	1109.50	0.260
16	1.74e+07	1.24	6.81e+09	4.020	1693.88	0.221
32	1.74e+07	1.26	6.81e+09	2.478	2747.96	0.144
64	1.74e+07	1.31	6.81e+09	4.292	1586.49	0.117
matrix : HSCT2, N: 88404, NNZ_LowerA: 1.8166E+06						
2	1.80e+07	1.03	8.60e+09	21.681	396.72	0.818
4	1.80e+07	1.15	8.60e+09	13.562	634.22	0.697
8	1.80e+07	1.46	8.60e+09	8.426	1020.78	0.514
16	1.80e+07	1.47	8.60e+09	5.107	1684.18	0.209
32	1.81e+07	1.66	8.70e+09	4.156	2092.37	0.234
64	1.82e+07	1.76	8.82e+09	4.790	1841.19	0.686
matrix : COPTER2, N: 55476, NNZ_LowerA: 4.0771E+05						
2	9.62e+06	1.08	5.40e+09	14.901	362.22	0.742
4	9.62e+06	1.16	5.40e+09	8.224	656.70	0.366
8	9.63e+06	1.22	5.41e+09	5.444	994.32	0.322
16	9.64e+06	1.29	5.44e+09	3.076	1767.94	0.132
32	9.65e+06	1.34	5.46e+09	2.166	2518.98	0.118
64	9.67e+06	1.42	5.47e+09	3.688	1482.98	0.087
matrix : CUBE35, N: 42875, NNZ_LowerA: 1.6783E+05						
2	8.04e+06	1.03	6.67e+09	18.094	368.88	0.504
4	8.04e+06	1.08	6.68e+09	8.941	746.94	0.262
8	8.04e+06	1.13	6.68e+09	7.456	896.01	0.195
16	8.05e+06	1.19	6.69e+09	4.462	1499.46	0.192
32	8.05e+06	1.21	6.70e+09	2.728	2455.30	0.118
64	8.06e+06	1.22	6.71e+09	4.112	1630.67	0.120
matrix : FORT50, N: 21901, NNZ_LowerA: 8.6987E+04						
2	4.88e+05	1.01	2.21e+07	0.357	62.03	0.085
4	4.88e+05	1.12	2.21e+07	0.231	95.97	0.064
8	4.88e+05	1.16	2.21e+07	0.121	183.30	0.024
16	4.88e+05	1.20	2.21e+07	0.105	210.66	0.040
32	4.88e+05	1.25	2.21e+07	0.237	93.22	0.222
64	4.88e+05	1.44	2.21e+07	1.341	16.50	0.021
matrix : BCSSTK18, N: 11948, NNZ_LowerA: 8.0500E+04						
2	6.12e+05	1.30	1.02e+08	0.662	153.85	0.078
4	6.12e+05	1.64	1.02e+08	0.411	248.02	0.034
8	6.14e+05	2.02	1.03e+08	0.268	383.12	0.026
16	6.17e+05	2.14	1.04e+08	0.202	516.81	0.020
32	6.20e+05	2.16	1.06e+08	0.326	324.30	0.021
64	6.26e+05	2.29	1.07e+08	1.589	67.31	0.038

TABLE 3

Performance for factorization and solve phases on Origin 2000 ($N < 100000$, Serial ordering and nrhs = 1)

np	NNZ_L	Imbal	FAC_OPC	Ntime	Nperf	Ttime
matrix : MDUAL2, N: 988605, NNZ_LowerA: 2.9357E+06						
4	1.62e+08	1.38	2.44e+11	553.923	440.98	6.779
8	1.62e+08	1.54	2.44e+11	305.034	800.79	4.728
16	1.62e+08	1.75	2.44e+11	142.065	1719.74	2.181
32	1.62e+08	1.93	2.44e+11	88.740	2754.25	1.566
64	1.63e+08	1.97	2.45e+11	48.500	5042.77	0.734
matrix : AORTA, N: 577771, NNZ_LowerA: 1.7067E+06						
2	7.56e+07	1.40	7.91e+10	300.726	263.00	6.139
4	7.56e+07	1.49	7.91e+10	166.409	475.28	3.580
8	7.57e+07	1.98	7.96e+10	119.636	665.21	2.031
16	7.60e+07	2.07	8.02e+10	51.922	1544.63	1.266
32	7.61e+07	2.07	8.05e+10	29.965	2687.14	0.767
64	7.62e+07	2.16	8.07e+10	22.628	3566.88	0.437
matrix : AUTO, N: 448695, NNZ_LowerA: 3.7633E+06						
8	2.36e+08	1.72	5.24e+11	937.230	559.24	5.606
16	2.36e+08	1.75	5.24e+11	350.055	1497.55	2.472
32	2.36e+08	1.80	5.24e+11	194.108	2701.02	1.785
64	2.36e+08	1.84	5.24e+11	109.526	4787.60	1.177
matrix : TRAIN, N: 390673, NNZ_LowerA: 1.5426E+06						
8	1.57e+08	1.26	4.33e+11	471.821	918.46	4.309
16	1.57e+08	1.31	4.34e+11	228.630	1897.21	2.240
32	1.57e+08	1.32	4.34e+11	117.561	3693.08	1.371
64	1.57e+08	1.33	4.35e+11	74.460	5835.52	0.747
matrix : CUBE65, N: 274625, NNZ_LowerA: 1.0858E+06						
4	1.17e+08	1.06	3.44e+11	610.914	562.66	3.996
8	1.17e+08	1.08	3.44e+11	328.662	1046.25	3.050
16	1.17e+08	1.16	3.44e+11	195.840	1756.90	1.790
32	1.17e+08	1.19	3.44e+11	94.217	3654.56	0.978
64	1.17e+08	1.22	3.44e+11	51.776	6653.68	0.536
matrix : TORSO, N: 201142, NNZ_LowerA: 1.6811E+06						
2	8.75e+07	1.10	1.43e+11	518.368	275.83	5.209
4	8.75e+07	1.15	1.43e+11	240.890	593.70	3.623
8	8.75e+07	1.30	1.43e+11	183.453	779.72	2.023
16	8.75e+07	1.40	1.43e+11	84.933	1686.32	1.175
32	8.76e+07	1.49	1.43e+11	46.889	3055.85	0.634
64	8.76e+07	1.49	1.43e+11	29.251	4901.31	0.365
matrix : CUBE50, N: 125000, NNZ_LowerA: 4.9250E+05						
2	3.89e+07	1.06	6.89e+10	200.402	343.84	2.380
4	3.90e+07	1.06	6.90e+10	136.814	504.53	1.873
8	3.90e+07	1.08	6.92e+10	55.284	1252.02	0.687
16	3.91e+07	1.15	6.94e+10	34.024	2040.64	0.518
32	3.91e+07	1.16	6.96e+10	17.899	3886.93	0.348
64	3.92e+07	1.17	6.97e+10	13.462	5175.73	0.246

TABLE 4

Performance for factorization and solve phases on Origin 2000 ($N > 100000$, Serial ordering and nrhs = 1)

np	NNZ_L	Imbal	FAC_OPC	Ntime	Nperf	Ttime
matrix : S3DKQ4M2, N: 90449, NNZ_LowerA: 2.4600E+06						
2	1.84e+07	1.01	7.63e+09	18.913	403.44	0.817
4	1.92e+07	1.07	8.91e+09	11.907	747.94	0.861
8	2.06e+07	1.31	1.09e+10	9.190	1186.06	0.488
16	2.23e+07	1.35	1.32e+10	7.202	1827.02	0.276
32	2.26e+07	1.59	1.31e+10	4.827	2709.90	0.189
64	2.42e+07	1.40	1.51e+10	3.873	3906.91	0.108
matrix : HSCT2, N: 88404, NNZ_LowerA: 1.8166E+06						
2	1.88e+07	1.03	9.63e+09	23.208	415.14	0.822
4	1.91e+07	1.15	1.02e+10	15.084	678.01	0.779
8	2.01e+07	1.57	1.17e+10	11.986	975.01	0.529
16	2.11e+07	2.01	1.31e+10	8.522	1533.79	0.281
32	2.16e+07	1.90	1.34e+10	4.734	2820.28	0.165
64	2.36e+07	2.05	1.58e+10	4.669	3383.89	0.129
matrix : COPTER2, N: 55476, NNZ_LowerA: 4.0771E+05						
2	9.98e+06	1.03	6.01e+09	15.463	388.53	0.557
4	1.03e+07	1.22	6.53e+09	13.158	496.07	0.550
8	1.05e+07	1.20	6.72e+09	5.552	1210.28	0.214
16	1.11e+07	1.49	7.49e+09	4.077	1837.73	0.178
32	1.19e+07	1.57	8.46e+09	3.822	2212.41	0.141
64	1.23e+07	1.60	8.81e+09	2.815	3129.11	0.092
matrix : CUBE35, N: 42875, NNZ_LowerA: 1.6783E+05						
2	8.64e+06	1.00	7.64e+09	20.578	371.24	0.516
4	8.91e+06	1.08	8.02e+09	10.840	739.81	0.285
8	9.26e+06	1.12	8.39e+09	6.812	1231.72	0.249
16	1.02e+07	1.15	9.38e+09	5.141	1824.62	0.145
32	1.05e+07	1.20	9.53e+09	2.939	3244.02	0.110
64	1.07e+07	1.27	9.37e+09	2.447	3829.95	0.101
matrix : FORT50, N: 21901, NNZ_LowerA: 8.6987E+04						
2	4.91e+05	1.03	2.25e+07	0.352	64.09	0.089
4	5.03e+05	1.17	2.56e+07	0.207	123.60	0.047
8	4.91e+05	1.38	2.25e+07	0.113	198.93	0.027
16	5.26e+05	1.40	2.86e+07	0.090	317.64	0.034
32	5.32e+05	1.35	2.91e+07	0.102	284.01	0.022
64	5.32e+05	1.76	2.83e+07	0.137	206.55	0.018
matrix : BCSSTK18, N: 11948, NNZ_LowerA: 8.0500E+04						
2	5.87e+05	1.23	8.84e+07	0.602	146.84	0.061
4	6.60e+05	1.86	1.21e+08	0.494	244.68	0.034
8	7.32e+05	1.71	1.49e+08	0.314	473.95	0.022
16	7.31e+05	1.66	1.44e+08	0.214	673.12	0.019
32	8.36e+05	1.97	1.83e+08	0.185	989.09	0.019

Table 5

Performance for factorization and solve phases on Origin 2000 (N < 100000, Parallel ordering and nrhs = 1)

np	NNZ_L	Imbal	FAC_OPC	Ntime	Nperf	Ttime
matrix : MDUAL2, N: 988605, NNZ_LowerA: 2.9357E+06						
4	1.75e+08	1.26	3.15e+11	722.282	435.48	7.390
8	1.88e+08	1.47	3.73e+11	445.339	838.48	5.109
16	2.05e+08	1.20	4.50e+11	229.390	1963.38	4.217
32	2.45e+08	1.37	6.63e+11	186.560	3553.50	2.139
64	2.18e+08	1.73	4.72e+11	81.121	5823.25	1.055
matrix : AORTA, N: 577771, NNZ_LowerA: 1.7067E+06						
2	7.30e+07	1.38	7.35e+10	258.867	284.01	6.415
4	7.65e+07	1.49	8.12e+10	163.594	496.27	3.844
8	7.52e+07	1.72	7.93e+10	97.519	812.87	1.847
16	7.90e+07	2.09	8.86e+10	67.405	1314.66	1.537
32	8.43e+07	2.04	9.83e+10	36.601	2686.69	0.993
64	8.98e+07	2.18	1.08e+11	24.136	4474.85	0.394
matrix : AUTO, N: 448695, NNZ_LowerA: 3.7633E+06						
16	2.64e+08	1.90	7.04e+11	477.660	1474.33	3.054
32	2.71e+08	1.84	7.24e+11	271.149	2671.20	1.965
64	2.82e+08	1.99	7.65e+11	147.678	5177.97	1.064
matrix : TRAIN, N: 390673, NNZ_LowerA: 1.5426E+06						
8	1.69e+08	1.24	5.07e+11	722.349	702.51	4.199
16	1.84e+08	1.31	5.74e+11	283.915	2022.30	2.156
32	1.92e+08	1.38	6.05e+11	168.887	3581.54	1.784
64	2.00e+08	1.21	6.24e+11	96.203	6485.72	1.002
matrix : CUBE65, N: 274625, NNZ_LowerA: 1.0858E+06						
4	1.26e+08	1.09	3.83e+11	763.093	501.36	4.651
8	1.29e+08	1.19	3.96e+11	437.912	904.92	3.170
16	1.36e+08	1.35	4.15e+11	251.764	1648.37	1.859
32	1.40e+08	1.19	4.26e+11	116.007	3673.95	0.986
64	1.45e+08	1.25	4.31e+11	62.771	6867.48	0.536
matrix : TORSO, N: 201142, NNZ_LowerA: 1.6811E+06						
2	8.96e+07	1.03	1.52e+11	490.762	310.38	5.148
4	9.68e+07	1.05	1.88e+11	352.899	532.72	3.289
8	9.41e+07	1.42	1.75e+11	204.694	855.68	2.492
16	9.75e+07	1.41	1.87e+11	107.421	1744.84	1.202
32	1.04e+08	1.93	2.11e+11	74.896	2810.68	0.778
64	1.08e+08	1.78	2.24e+11	46.570	4818.53	0.396
matrix : CUBE50, N: 125000, NNZ_LowerA: 4.9250E+05						
2	3.90e+07	1.00	6.91e+10	195.160	354.20	2.006
4	3.93e+07	1.10	7.03e+10	126.632	554.91	1.643
8	4.38e+07	1.13	8.04e+10	64.184	1252.52	1.001
16	4.61e+07	1.26	8.47e+10	36.075	2348.41	0.584
32	4.80e+07	1.27	8.76e+10	22.470	3897.91	0.392
64	4.83e+07	1.26	8.59e+10	14.166	6065.18	0.239

TABLE 6

Performance for factorization and solve phases on Origin 2000 ($N > 100000$, Parallel ordering and $nrhs = 1$)

np	NNZ_L	Imbal	FAC_OPC	Ntime	Nperf	Ttime
matrix : S3DKQ4M2, N: 90449, NNZ_LowerA: 2.4600E+06						
2	1.74e+07	1.00	6.81e+09	14.083	483.52	0.451
4	1.74e+07	1.02	6.81e+09	7.245	939.88	0.234
8	1.74e+07	1.17	6.81e+09	4.128	1649.66	0.131
16	1.74e+07	1.24	6.81e+09	2.445	2784.95	0.086
32	1.74e+07	1.26	6.81e+09	1.376	4949.22	0.063
64	1.74e+07	1.31	6.81e+09	0.857	7949.59	0.053
128	1.74e+07	1.33	6.81e+09	0.497	13692.14	0.042
256	1.74e+07	1.40	6.81e+09	0.369	18479.03	0.079
matrix : HSCT2, N: 88404, NNZ_LowerA: 1.8166E+06						
2	1.80e+07	1.03	8.60e+09	18.376	468.06	0.504
4	1.80e+07	1.15	8.60e+09	10.701	803.78	0.266
8	1.80e+07	1.46	8.60e+09	7.129	1206.54	0.151
16	1.80e+07	1.47	8.60e+09	4.361	1972.46	0.084
32	1.81e+07	1.66	8.70e+09	1.968	4419.81	0.060
64	1.82e+07	1.76	8.82e+09	1.190	7410.03	0.045
128	1.83e+07	1.78	8.92e+09	0.717	12441.58	0.072
256	1.84e+07	1.80	8.98e+09	0.473	18987.09	0.196
matrix : COPTER2, N: 55476, NNZ_LowerA: 4.0771E+05						
2	9.62e+06	1.08	5.40e+09	11.621	464.44	0.390
4	9.62e+06	1.16	5.40e+09	6.113	883.56	0.212
8	9.63e+06	1.22	5.41e+09	3.831	1413.07	0.130
16	9.64e+06	1.29	5.44e+09	2.820	1928.30	0.071
32	9.65e+06	1.34	5.46e+09	1.239	4402.68	0.053
64	9.67e+06	1.42	5.47e+09	0.761	7187.79	0.040
128	9.68e+06	1.44	5.48e+09	0.496	11035.01	0.048
256	9.69e+06	1.45	5.48e+09	0.398	13782.89	0.094

TABLE 7

Performance for factorization and solve phases on Cray T3E-1200 ($N < 100000$, Serial ordering and $nrhs = 1$)

np	NNZ_L	Imbal	FAC_OPC	Ntime	Nperf	Ttime
matrix : CUBE35, N: 42875, NNZ_LowerA: 1.6783E+05						
2	8.04e+06	1.03	6.67e+09	13.223	504.76	0.403
4	8.04e+06	1.08	6.68e+09	7.009	952.79	0.215
8	8.04e+06	1.13	6.68e+09	3.924	1702.33	0.123
16	8.05e+06	1.19	6.69e+09	2.275	2941.31	0.078
32	8.05e+06	1.21	6.70e+09	1.439	4655.27	0.057
64	8.06e+06	1.22	6.71e+09	0.806	8324.08	0.047
128	8.07e+06	1.23	6.71e+09	0.537	12493.35	0.173
256	8.09e+06	1.23	6.72e+09	0.414	16216.24	0.322
matrix : FORT50, N: 21901, NNZ_LowerA: 8.6987E+04						
2	4.88e+05	1.01	2.21e+07	0.308	71.71	0.108
4	4.88e+05	1.12	2.21e+07	0.169	131.12	0.057
8	4.88e+05	1.16	2.21e+07	0.098	225.43	0.030
16	4.88e+05	1.20	2.21e+07	0.062	355.04	0.018
32	4.88e+05	1.25	2.21e+07	0.049	452.43	0.012
64	4.88e+05	1.44	2.21e+07	0.052	428.26	0.009
128	4.89e+05	1.45	2.22e+07	0.073	305.64	0.012
256	4.90e+05	1.49	2.23e+07	0.110	202.66	0.115
matrix : BCSSTK18, N: 11948, NNZ_LowerA: 8.0500E+04						
2	6.12e+05	1.30	1.02e+08	0.510	199.54	0.073
4	6.12e+05	1.64	1.02e+08	0.309	330.13	0.040
8	6.14e+05	2.02	1.03e+08	0.212	484.07	0.023
16	6.17e+05	2.14	1.04e+08	0.154	677.57	0.017
32	6.20e+05	2.16	1.06e+08	0.113	938.34	0.016
64	6.26e+05	2.29	1.07e+08	0.102	1051.64	0.015
128	6.30e+05	2.35	1.08e+08	0.107	1011.43	0.042
256	6.34e+05	2.38	1.08e+08	0.153	705.07	0.024

TABLE 8

Performance for factorization and solve phases on Cray T3E-1200 ($N < 100000$, Serial ordering and $nrhs = 1$)

np	NNZ_L	Imbal	FAC_OPC	Ntime	Nperf	Ttime
matrix : MDUAL2, N: 988605, NNZ_LowerA: 2.9357E+06						
32	1.62e+08	1.93	2.44e+11	46.638	5240.66	0.620
64	1.63e+08	1.97	2.45e+11	26.525	9220.47	0.349
128	1.63e+08	2.01	2.45e+11	14.215	17218.57	0.233
256	1.63e+08	2.01	2.45e+11	9.306	26319.99	0.332
matrix : AORTA, N: 577771, NNZ_LowerA: 1.7067E+06						
8	7.57e+07	1.98	7.96e+10	92.621	859.22	1.271
16	7.60e+07	2.07	8.02e+10	35.023	2289.92	0.660
32	7.61e+07	2.07	8.05e+10	19.847	4057.07	0.369
64	7.62e+07	2.16	8.07e+10	12.033	6707.30	0.245
128	7.63e+07	2.18	8.08e+10	6.650	12152.02	0.279
256	7.63e+07	2.21	8.09e+10	3.678	21985.12	0.221
matrix : AUTO, N: 448695, NNZ_LowerA: 3.7633E+06						
64	2.36e+08	1.84	5.24e+11	56.223	9326.57	0.325
128	2.36e+08	1.85	5.24e+11	27.898	18797.40	0.285
256	2.36e+08	1.86	5.24e+11	18.358	28567.93	0.951
matrix : TRAIN, N: 390673, NNZ_LowerA: 1.5426E+06						
32	1.57e+08	1.32	4.34e+11	64.800	6700.06	0.438
64	1.57e+08	1.33	4.35e+11	33.856	12834.17	0.272
128	1.57e+08	1.34	4.35e+11	20.003	21738.39	0.397
256	1.57e+08	1.34	4.35e+11	11.354	38318.00	0.349
matrix : CUBE65, N: 274625, NNZ_LowerA: 1.0858E+06						
32	1.17e+08	1.19	3.44e+11	50.284	6847.52	0.351
64	1.17e+08	1.22	3.45e+11	27.846	12371.72	0.296
128	1.17e+08	1.22	3.45e+11	14.107	24429.80	0.389
256	1.17e+08	1.22	3.45e+11	8.622	39981.14	1.271
matrix : TORSO, N: 201142, NNZ_LowerA: 1.6811E+06						
16	8.75e+07	1.40	1.43e+11	45.502	3147.66	0.404
32	8.76e+07	1.49	1.43e+11	28.479	5031.27	0.273
64	8.76e+07	1.49	1.43e+11	13.445	10663.10	0.162
128	8.77e+07	1.51	1.43e+11	7.239	19814.06	0.218
256	8.77e+07	1.52	1.43e+11	4.648	30872.54	1.097
matrix : CUBE50, N: 125000, NNZ_LowerA: 4.9250E+05						
4	3.90e+07	1.06	6.90e+10	59.939	1151.61	0.764
8	3.90e+07	1.08	6.92e+10	32.435	2134.03	0.407
16	3.91e+07	1.15	6.94e+10	17.875	3884.17	0.238
32	3.91e+07	1.16	6.96e+10	9.910	7020.09	0.156
64	3.92e+07	1.17	6.97e+10	5.412	12874.28	0.121
128	3.92e+07	1.18	6.97e+10	3.117	22376.83	0.372
256	3.93e+07	1.18	6.98e+10	1.989	35087.72	0.324

TABLE 9

Performance for factorization and solve phases on Cray T3E-1200 ($N > 100000$, Serial ordering and nrhs = 1)

np	NNZ_L	Imbal	FAC_OPC	Ntime	Nperf	Ttime
matrix : S3DKQ4M2, N: 90449, NNZ_LowerA: 2.4600E+06						
2	1.84e+07	1.01	7.63e+09	17.482	436.46	0.491
4	1.92e+07	1.07	8.91e+09	10.973	811.60	0.263
8	2.06e+07	1.31	1.09e+10	7.008	1555.33	0.162
16	2.23e+07	1.35	1.32e+10	4.207	3128.08	0.094
32	2.26e+07	1.59	1.31e+10	2.786	4695.26	0.074
64	2.42e+07	1.40	1.51e+10	1.549	9770.35	0.055
128	2.23e+07	1.56	1.20e+10	0.866	13795.57	0.155
256	2.54e+07	1.65	1.54e+10	0.711	21633.39	0.099
matrix : HSCT2, N: 88404, NNZ_LowerA: 1.8166E+06						
2	1.88e+07	1.03	9.63e+09	20.723	464.92	0.507
4	1.91e+07	1.15	1.02e+10	11.438	894.11	0.277
8	2.01e+07	1.57	1.17e+10	8.164	1431.49	0.171
16	2.11e+07	2.01	1.31e+10	6.022	2170.37	0.105
32	2.16e+07	1.90	1.34e+10	3.190	4185.55	0.072
64	2.36e+07	2.05	1.58e+10	2.207	7156.60	0.062
128	2.21e+07	2.37	1.35e+10	1.337	10078.15	0.120
256	2.44e+07	2.58	1.57e+10	0.977	16068.43	0.169
matrix : COPTER2, N: 55476, NNZ_LowerA: 4.0771E+05						
2	9.98e+06	1.03	6.01e+09	11.861	506.51	0.407
4	1.03e+07	1.22	6.53e+09	7.659	852.23	0.221
8	1.05e+07	1.20	6.72e+09	4.872	1379.30	0.129
16	1.11e+07	1.49	7.49e+09	2.879	2602.55	0.078
32	1.19e+07	1.57	8.46e+09	1.817	4652.59	0.060
64	1.23e+07	1.60	8.81e+09	1.146	7684.97	0.048
128	1.19e+07	2.02	8.01e+09	0.806	9945.28	0.080
256	1.22e+07	1.36	8.06e+09	0.466	17297.36	0.298

TABLE 10

Performance for factorization and solve phases on Cray T3E-1200 ($N < 100000$, Parallel ordering and $nrhs = 1$)

np	NNZ_L	Imbal	FAC_OPC	Ntime	Nperf	Ttime
matrix : CUBE35, N: 42875, NNZ_LowerA: 1.6783E+05						
2	8.64e+06	1.00	7.64e+09	13.985	546.26	0.413
4	8.91e+06	1.08	8.02e+09	10.301	778.53	0.230
8	9.26e+06	1.12	8.39e+09	5.538	1515.19	0.129
16	1.02e+07	1.15	9.38e+09	2.908	3225.44	0.084
32	1.05e+07	1.20	9.53e+09	1.673	5697.62	0.059
64	1.07e+07	1.27	9.37e+09	1.076	8713.59	0.048
128	1.08e+07	1.36	9.33e+09	0.733	12737.00	0.072
256	1.10e+07	1.33	9.61e+09	0.545	17631.78	0.403
matrix : FORT50, N: 21901, NNZ_LowerA: 8.6987E+04						
2	4.91e+05	1.03	2.25e+07	0.306	73.61	0.114
4	5.03e+05	1.17	2.56e+07	0.173	147.91	0.062
8	4.91e+05	1.38	2.25e+07	0.097	231.16	0.033
16	5.26e+05	1.40	2.86e+07	0.069	411.11	0.019
32	5.32e+05	1.35	2.91e+07	0.053	551.44	0.012
64	5.32e+05	1.76	2.83e+07	0.054	527.83	0.009
128	5.33e+05	1.55	2.80e+07	0.062	454.98	0.027
256	5.08e+05	1.61	2.47e+07	0.069	359.67	0.067
matrix : BCSSTK18, N: 11948, NNZ_LowerA: 8.0500E+04						
2	5.87e+05	1.23	8.84e+07	0.512	172.59	0.072
4	6.60e+05	1.86	1.21e+08	0.419	288.25	0.039
8	7.32e+05	1.71	1.49e+08	0.266	559.90	0.025
16	7.31e+05	1.66	1.44e+08	0.159	905.27	0.017
32	8.36e+05	1.97	1.83e+08	0.141	1299.53	0.014
64	8.50e+05	2.22	1.86e+08	0.122	1523.63	0.017
128	6.80e+05	2.44	1.12e+08	0.087	1282.71	0.024
256	6.34e+05	2.38	1.08e+08	0.153	705.65	0.014

Table 11

Performance for factorization and solve phases on Cray T3E-1200 ($N < 100000$, Parallel ordering and nrhs = 1)

np	NNZ_L	Imbal	FAC_OPC	Ntime	Nperf	Ttime
matrix : CUBE100 N: 1000000, NNZ_LowerA: 3.9700E+06						
256	9.10e+08	1.37	6.13e+12	120.090	51026.96	0.996
matrix : MDUAL2, N: 988605, NNZ_LowerA: 2.9357E+06						
64	2.18e+08	1.73	4.72e+11	49.410	9560.52	0.455
128	2.02e+08	1.56	3.76e+11	19.600	19160.32	1.303
256	2.14e+08	1.64	4.10e+11	11.193	36608.05	0.257
matrix : AORTA, N: 577771, NNZ_LowerA: 1.7067E+06						
8	7.52e+07	1.72	7.93e+10	54.644	1450.65	1.332
16	7.90e+07	2.09	8.86e+10	38.043	2329.34	0.721
32	8.43e+07	2.04	9.83e+10	22.066	4456.27	0.394
64	8.98e+07	2.18	1.08e+11	12.431	8688.20	0.254
128	9.08e+07	2.37	1.04e+11	7.964	13045.50	0.615
256	9.51e+07	2.09	1.08e+11	4.515	24030.90	0.366
matrix : AUTO, N: 448695, NNZ_LowerA: 3.7633E+06						
64	2.82e+08	1.99	7.65e+11	69.881	10942.54	0.356
128	2.71e+08	1.86	6.97e+11	34.189	20377.36	0.674
256	2.76e+08	1.88	6.88e+11	20.246	33970.49	0.613
matrix : TRAIN, N: 390673, NNZ_LowerA: 1.5426E+06						
32	1.92e+08	1.38	6.05e+11	88.383	6843.84	0.499
64	2.00e+08	1.21	6.24e+11	47.846	13040.55	0.374
128	2.03e+08	1.51	6.21e+11	26.094	23813.96	0.563
256	2.04e+08	1.49	5.90e+11	13.948	42285.06	0.510
matrix : CUBE65, N: 274625, NNZ_LowerA: 1.0858E+06						
32	1.40e+08	1.19	4.26e+11	60.234	7075.78	0.438
64	1.45e+08	1.25	4.31e+11	35.762	12054.11	0.318
128	1.48e+08	1.27	4.33e+11	19.082	22716.28	0.343
256	1.51e+08	1.28	4.32e+11	9.965	43315.06	0.496
matrix : TORSO, N: 201142, NNZ_LowerA: 1.6811E+06						
16	9.75e+07	1.41	1.87e+11	56.129	3339.32	0.405
32	1.04e+08	1.93	2.11e+11	40.147	5243.39	0.286
64	1.08e+08	1.78	2.24e+11	26.953	8325.65	0.194
128	9.96e+07	1.60	1.78e+11	9.191	19386.77	0.313
256	1.04e+08	1.41	1.93e+11	5.451	35472.91	0.722
matrix : CUBE50, N: 125000, NNZ_LowerA: 4.9250E+05						
4	3.93e+07	1.10	7.03e+10	67.295	1044.21	0.828
8	4.38e+07	1.13	8.04e+10	39.360	2042.46	0.440
16	4.61e+07	1.26	8.47e+10	22.137	3827.07	0.277
32	4.80e+07	1.27	8.76e+10	12.546	6981.23	0.174
64	4.83e+07	1.26	8.59e+10	7.189	11951.77	0.123
128	4.94e+07	1.36	8.67e+10	4.144	20919.51	0.174
256	5.06e+07	1.36	8.74e+10	2.523	34633.05	0.338

Table 12

Performance for factorization and solve phases on Cray T3E-1200 ($N > 100000$, Parallel ordering and nrhs = 1)

np	Imbal	FAC_OPC	Otime	Ytime	Ntime	Nperf	Ttime	PSPtime
matrix : S3DKQ4M2, N: 90449, NNZ_LowerA: 2.4600E+06								
2	1.01	7.63e+09	8.31	2.46	18.913	403.44	0.883	30.567
4	1.07	8.91e+09	5.62	3.37	11.907	747.94	0.929	21.831
8	1.31	1.09e+10	4.27	2.10	9.190	1186.06	0.553	16.114
16	1.35	1.32e+10	3.77	1.27	7.202	1827.02	0.334	12.583
32	1.59	1.31e+10	3.73	1.11	4.827	2709.90	0.273	9.935
64	1.40	1.51e+10	16.02	2.16	3.873	3906.91	0.404	22.449
matrix : HSCT2, N: 88404, NNZ_LowerA: 1.8166E+06								
2	1.03	9.63e+09	8.13	2.01	23.208	415.14	0.901	34.253
4	1.15	1.02e+10	5.34	1.33	15.084	678.01	0.830	22.584
8	1.57	1.17e+10	3.31	1.11	11.986	975.01	0.587	16.997
16	2.01	1.31e+10	3.97	0.97	8.522	1533.79	0.345	13.807
32	1.90	1.34e+10	4.75	0.60	4.734	2820.28	0.244	10.327
64	2.05	1.58e+10	18.01	2.01	4.669	3383.89	0.297	24.983
matrix : COPTER2, N: 55476, NNZ_LowerA: 4.0771E+05								
2	1.03	6.01e+09	4.91	0.62	15.463	388.53	0.600	21.586
4	1.22	6.53e+09	4.11	0.48	13.158	496.07	0.608	18.361
8	1.20	6.72e+09	2.11	0.28	5.552	1210.28	0.243	8.186
16	1.49	7.49e+09	2.08	0.29	4.077	1837.73	0.221	6.668
32	1.57	8.46e+09	7.72	0.68	3.822	2212.41	0.202	12.424
64	1.60	8.81e+09	14.75	1.00	2.815	3129.11	0.172	18.735
matrix : CUBE35, N: 42875, NNZ_LowerA: 1.6783E+05								
2	1.00	7.64e+09	2.24	0.34	20.578	371.24	0.550	23.716
4	1.08	8.02e+09	1.34	0.23	10.840	739.81	0.311	12.722
8	1.12	8.39e+09	1.15	0.24	6.812	1231.72	0.280	8.485
16	1.15	9.38e+09	1.34	0.18	5.141	1824.62	0.176	6.845
32	1.20	9.53e+09	2.90	0.24	2.939	3244.02	0.151	6.232
64	1.27	9.37e+09	16.98	0.97	2.447	3829.95	0.340	20.736
matrix : FORT50, N: 21901, NNZ_LowerA: 8.6987E+04								
2	1.03	2.25e+07	0.92	0.12	0.352	64.09	0.106	1.500
4	1.17	2.56e+07	0.55	0.08	0.207	123.60	0.059	0.896
8	1.38	2.25e+07	0.47	0.05	0.113	198.93	0.037	0.672
16	1.40	2.86e+07	0.84	0.06	0.090	317.64	0.052	1.046
32	1.35	2.91e+07	2.00	0.11	0.102	284.01	0.065	2.274
64	1.76	2.83e+07	17.46	0.48	0.137	206.55	0.081	18.158
matrix : BCSSTK18, N: 11948, NNZ_LowerA: 8.0500E+04								
2	1.23	8.84e+07	0.88	0.11	0.602	146.84	0.071	1.666
4	1.86	1.21e+08	0.56	0.07	0.494	244.68	0.041	1.163
8	1.71	1.49e+08	0.53	0.04	0.314	473.95	0.029	0.916
16	1.66	1.44e+08	0.95	0.05	0.214	673.12	0.028	1.236
32	1.97	1.83e+08	2.03	0.09	0.185	989.09	0.041	2.347

Table 13

Performance for All Phases on Origin 2000 ($N < 100000$, Parallel ordering and nrhs = 1)

np	Imbal	FAC_OPC	Otime	Ytime	Ntime	Nperf	Ttime	PSPtime
matrix : MDUAL2, N: 988605, NNZ_LowerA: 2.9357E+06								
4	1.26	3.15e+11	54.20	7.03	722.282	435.48	8.373	791.890
8	1.47	3.73e+11	33.73	6.27	445.339	838.48	5.868	491.216
16	1.20	4.50e+11	20.43	6.47	229.390	1963.38	5.107	261.400
32	1.37	6.63e+11	14.24	5.23	186.560	3553.50	2.810	208.835
64	1.73	4.72e+11	37.42	13.47	81.121	5823.25	2.950	134.968
matrix : AORTA, N: 577771, NNZ_LowerA: 1.7067E+06								
2	1.38	7.35e+10	35.28	3.93	258.867	284.01	6.964	305.039
4	1.49	8.12e+10	20.76	3.95	163.594	496.27	4.302	192.608
8	1.72	7.93e+10	12.23	2.49	97.519	812.87	2.236	114.479
16	2.09	8.86e+10	8.71	2.47	67.405	1314.66	1.991	80.575
32	2.04	9.83e+10	9.25	3.54	36.601	2686.69	1.602	50.993
64	2.18	1.08e+11	21.74	7.33	24.136	4474.85	1.139	54.348
matrix : AUTO, N: 448695, NNZ_LowerA: 3.7633E+06								
16	1.90	7.04e+11	16.38	3.14	477.660	1474.33	3.318	500.499
32	1.84	7.24e+11	11.51	2.73	271.149	2671.20	2.362	287.743
64	1.99	7.65e+11	25.01	5.66	147.678	5177.97	1.684	180.033
matrix : TRAIN, N: 390673, NNZ_LowerA: 1.5426E+06								
8	1.24	5.07e+11	9.75	2.49	722.349	702.51	4.516	739.102
16	1.31	5.74e+11	6.06	2.73	283.915	2022.30	2.395	295.101
32	1.38	6.05e+11	8.13	2.53	168.887	3581.54	2.057	181.604
64	1.21	6.24e+11	26.20	5.96	96.203	6485.72	1.609	129.968
matrix : CUBE65, N: 274625, NNZ_LowerA: 1.0858E+06								
4	1.09	3.83e+11	12.45	2.82	763.093	501.36	4.882	783.247
8	1.19	3.96e+11	6.12	1.91	437.912	904.92	3.354	449.298
16	1.35	4.15e+11	5.63	1.78	251.764	1648.37	2.043	261.219
32	1.19	4.26e+11	5.35	2.12	116.007	3673.95	1.182	124.659
64	1.25	4.31e+11	15.34	4.91	62.771	6867.48	0.865	83.885
matrix : TORSO, N: 201142, NNZ_LowerA: 1.6811E+06								
2	1.03	1.52e+11	28.41	3.86	490.762	310.38	5.358	528.386
4	1.05	1.88e+11	12.12	2.16	352.899	532.72	3.421	370.598
8	1.42	1.75e+11	10.64	1.78	204.694	855.68	2.682	219.794
16	1.41	1.87e+11	6.76	1.16	107.421	1744.84	1.313	116.659
32	1.93	2.11e+11	7.35	1.18	74.896	2810.68	0.902	84.329
64	1.78	2.24e+11	29.35	3.25	46.570	4818.53	0.815	79.980
matrix : CUBE50, N: 125000, NNZ_LowerA: 4.9250E+05								
2	1.00	6.91e+10	7.09	1.16	195.160	354.20	2.114	205.533
4	1.10	7.03e+10	4.25	0.93	126.632	554.91	1.740	133.550
8	1.13	8.04e+10	2.28	0.63	64.184	1252.52	1.087	68.181
16	1.26	8.47e+10	2.31	0.70	36.075	2348.41	0.647	39.733
32	1.27	8.76e+10	3.83	0.87	22.470	3897.91	0.532	27.701
64	1.26	8.59e+10	14.47	1.90	14.166	6065.18	0.478	31.015

TABLE 14

Performance for All Phases on Origin 2000 ($N > 100000$, Parallel ordering and nrhs = 1)