A Fine-Grain Hypergraph Model for 2D Decomposition of Sparse Matrices*

Ümit V. Çatalyürek
Dept. of Pathology, Division of Informatics
Johns Hopkins Medical Institutions
Baltimore, MD 21287
umit@jhmi.edu

Cevdet Aykanat
Computer Engineering Department
Bilkent University
Ankara, 06533 Turkey
aykanat@cs.bilkent.edu.tr

Abstract

We propose a new hypergraph model for the decomposition of irregular computational domains. This work focuses on the decomposition of sparse matrices for parallel matrix-vector multiplication. However, the proposed model can also be used to decompose computational domains of other parallel reduction problems. We propose a "fine-grain" hypergraph model for two-dimensional decomposition of sparse matrices. In the proposed fine-grain hypergraph model, vertices represent nonzeros and hyperedges represent sparsity patterns of rows and columns of the matrix. By partitioning the fine-grain hypergraph into equally weighted vertex parts (processors) so that hyperedges are split among as few processors as possible, the model correctly minimizes communication volume while maintaining computational load balance. Experimental results on a wide range of realistic sparse matrices confirm the validity of the proposed model, by achieving up to 50 percent better decompositions than the existing models, in terms of total communication volume.

1 Introduction

Repeated matrix-vector multiplication $\mathbf{y} = \mathbf{A}\mathbf{x}$ that involves the same large, sparse, structurally symmetric or nonsymmetric square matrix $\mathbf{A}$ is the kernel operation in iterative solvers. These algorithms also involve linear operations on dense vectors. For efficient parallelization of these iterative algorithms, matrix $\mathbf{A}$ should be partitioned among processors in such a way that communication overhead is kept low while maintaining computational load balance. In order to avoid the communication of vector components during the linear vector operations, a symmetric partitioning scheme is adopted. That is, all vectors (including $\mathbf{x}$ and $\mathbf{y}$ vectors) used in the solver are divided conformally.

*This work is partially supported by Turkish Science and Research Council under grant EEEAG-199E013.

The standard graph partitioning approach has been widely used for one-dimensional (1D) decomposition of irregularly sparse matrices. In recent works, we [3, 4], and Hendrickson [9] mentioned the flaws and shortcomings of the standard graph-partitioning approach. In our recent works [3, 4], we proposed hypergraph-partitioning approach which correctly minimizes the communication volume in 1D matrix decomposition. Other recently proposed alternative models for 1D matrix decomposition were discussed in the excellent survey by Hendrickson and Kolda [10].

The literature that addresses 2D matrix decomposition is very rare. The 2D checkerboard decomposition schemes proposed by Hendrickson et al. [11] and Lewis and van de Geijn [15] are typically suitable for dense matrices or sparse matrices with structured nonzero patterns that are difficult to exploit. These schemes do not involve explicit effort towards reducing communication volume.

Parallel matrix-vector multiplication is one of the basic parallel reduction algorithms. Elements of $\mathbf{x}$ vector are the inputs of the reduction and elements of $\mathbf{y}$ vector are the outputs of the reduction. Matrix $\mathbf{A}$ corresponds to the mapping matrix from input elements to output elements. Hence, any technique used in the sparse matrix decomposition is also applicable to other reduction problems.

In this paper, we propose a fine-grain hypergraph-partitioning model for 2D decomposition of irregularly sparse matrices based on our previous work [2]. Vertices of the proposed fine-grain hypergraph correspond to the nonzeros of the matrix to model each scalar multiplication operation as an atomic task during the decomposition. Hyperedges of the fine-grain hypergraph correspond to columns and rows of the matrix to model the communication volume requirement of the expand and fold operations in the parallel matrix-vector multiplication. By partitioning the fine-grain hypergraph into equally weighted vertex parts (processors) so that hyperedges are split among as few processors as possible, the model correctly minimizes communication volume while maintaining computational load balance.

2 Preliminaries

A hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{N})$ is defined as a set of vertices $\mathcal{V}$ and a set of nets (hyperedges) $\mathcal{N}$ among those vertices. Every net $n_j \in \mathcal{N}$ is a subset of vertices, i.e., $n_j \subseteq \mathcal{V}$. The vertices in a net n_j are called its *pins* and denoted as $pins[n_j]$. The set of nets connected to a vertex v_i is denoted as $nets[v_i]$. Weights and costs can be assigned to the vertices and edges of the hypergraph, respectively. Let w_i and c_j denote the weight of vertex $v_i \in \mathcal{V}$ and the cost of net $n_j \in \mathcal{N}$, respectively.

$\Pi = \{\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_K\}$ is a *K-way partition* of $\mathcal{H}$ if the following conditions hold: each part $\mathcal{P}_k, 1 \leq k \leq K$, is a nonempty subset of $\mathcal{V}$, parts are pairwise disjoint ($\mathcal{P}_k \cap \mathcal{P}_\ell = \emptyset$ for all $1 \leq k < \ell \leq K$), and union of K parts is equal to $\mathcal{V}$ (i.e. $\bigcup_{k=1}^{K} \mathcal{P}_k = \mathcal{V}$). A partition is said to be balanced if each part $\mathcal{P}_k$ satisfies the *balance criterion*

$$W_k \leq W_{avg}(1+\varepsilon), \quad for\ k = 1, 2, \ldots, K. \qquad (1)$$

In (1), weight W_k of a part $\mathcal{P}_k$ is defined as the sum of the weights of the vertices in that part (i.e. $W_k = \sum_{v_i \in \mathcal{P}_k} w_i$), $W_{avg} = (\sum_{v_i \in \mathcal{V}} w_i)/K$ denotes the weight of each part under the perfect load balance condition, and ε represents the predetermined maximum imbalance ratio allowed.

In a partition Π of $\mathcal{H}$, a net that has at least one pin (vertex) in a part is said to *connect* that part. *Connectivity set* Λ_j of a net n_j is defined as the set of parts connected by n_j. *Connectivity* $\lambda_j = |\Lambda_j|$ of a net n_j denotes the number of parts connected by n_j. A net n_j is said to be *cut* if it connects more than one part (i.e. $\lambda_j > 1$), and *uncut* otherwise (i.e. $\lambda_j = 1$). The cut and uncut nets are also referred to here as *external* and *internal* nets, respectively. The set of external nets of a partition Π is denoted as $\mathcal{N}_E$. There are various *cutsize* definitions for representing the cost $\chi(\Pi)$ of a partition Π. Two relevant definitions are:

$$\chi(\Pi) = \sum_{n_j \in \mathcal{N}_E} c_j \qquad (2)$$

$$\chi(\Pi) = \sum_{n_j \in \mathcal{N}_E} c_j(\lambda_j - 1). \qquad (3)$$

In (2), the cutsize is equal to the sum of the costs of the cut nets. In (3), each cut net n_j contributes $c_j(\lambda_j - 1)$ to the cutsize. Hence, the hypergraph partitioning problem [14] can be defined as the task of dividing a hypergraph into two or more parts such that the cutsize is minimized, while a given balance criterion (1) among the part weights is maintained. The hypergraph partitioning problem is known to be NP-hard [14].

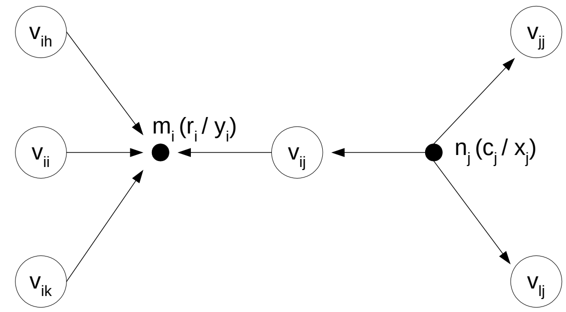

Figure 1. Dependency relation of 2D fine-grain hypergraph model

3 A Fine-grain Hypergraph Model

In this model, an $M \times M$ matrix **A** with Z nonzero elements is represented as a hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{N})$ with $|\mathcal{V}| = Z$ vertices and $|\mathcal{N}| = 2 \times M$ nets for 2D decomposition. There exists one vertex $v_{ij} \in \mathcal{V}$ corresponding to each nonzero a_{ij} in matrix **A**. For each row and for each column there exists a net in $\mathcal{N}$. For simplicity in the presentation let $\mathcal{N} = \mathcal{N}_\mathcal{R} \cup \mathcal{N}_\mathcal{C}$ such that $\mathcal{N}_\mathcal{R} = \{m_1, m_2, \ldots, m_M\}$ represents the set of nets corresponding to the rows and $\mathcal{N}_\mathcal{C} = \{n_1, n_2, \ldots, n_M\}$ represents the set of nets corresponding to the columns of matrix **A**. Net $m_j \subseteq \mathcal{V}$ contains the vertices corresponding to the nonzeros in row j, and net $n_j \subseteq \mathcal{V}$ contains the vertices corresponding to the nonzeros in column j. That is, $v_{ij} \in m_i$ and $v_{ij} \in n_j$ if and only if $a_{ij} \neq 0$. Note that each vertex v_{ij} is connected to exactly two nets. Each vertex $v_{ij} \in \mathcal{V}$ corresponds to the scalar multiplication operation $y_i^j = a_{ij} x_j$. Hence, each vertex $v_{ij} \in \mathcal{V}$ has unit computational weight $w_{ij} = 1$. The nets in $\mathcal{N}_\mathcal{C}$ represent the dependency relations of the scalar multiplication operations on the **x**-vector entries, that is, they model the expand operation in the pre communication phase. The nets in $\mathcal{N}_\mathcal{R}$ represent the dependency relations of the addition operations needed to accumulate the **y**-vector entries on the scalar multiplication results, in other words, they model the fold operation in the post communication phase. Hence, each column-net n_j denotes the set of scalar multiplication operations (vertices) that need x_j during pre communication, and each row-net m_i denotes the set of scalar multiplication results needed to accumulate y_i. Figure 1 illustrates the dependency relation view of 2D fine-grain model. As seen in this figure, column-net $n_j = \{v_{ij}, v_{jj}, v_{lj}\}$ of size 3 represents the dependency of atomic tasks v_{ij}, v_{jj}, v_{lj} to x_j because of the 3 multiplication operations $y_i^j = a_{ij} x_j$, $y_j^j = a_{jj} x_j$ and $y_l^j = a_{lj} x_j$. In this figure, row-net $m_i = \{v_{ih}, v_{ii}, v_{ik}, v_{ij}\}$ of size 4 represents the dependency of accumulating $y_i = y_i^h + y_i^i + y_i^k + y_i^j$ to the 4 partial y_i results $y_i^h = a_{ih} x_h$, $y_i^i = a_{ii} x_i$, $y_i^k = a_{ik} x_k$ and $y_i^j = a_{ij} x_j$.

By assigning unit costs to the nets (i.e. $c_j = 1$ for each net $n_j \in \mathcal{N}$), the proposed fine-grain hypergraph model reduces the 2D matrix decomposition problem for a parallel system with K processors to the K-way hypergraph partitioning problem according to the cutsize definition given in (3). Nets corresponding to rows of matrix (i.e. nets in $\mathcal{N}_\mathcal{R}$) model the communication volume requirement of folds, and nets corresponding to columns of matrix (i.e. nets in $\mathcal{N}_\mathcal{C}$) model the communication volume requirement of expands.

Consistency of the proposed hypergraph models for accurate representation of communication volume requirement while maintaining symmetric partitioning depends on the condition that "$v_{ii} \in pins[m_i]$ and $v_{ii} \in pins[n_i]$ for each row-net m_i and column-net n_i". Note that symmetric partitioning — having the same number of input and output elements and assigning the corresponding input and output elements to the same processor — may not be required in the other reduction problems hence in the decomposition of those problems we may not need such a consistency condition. That is, in the absence of symmetric partitioning requirement, the proposed model already achieves the accurate representation of communication volume requirement without consistency condition. In some other reduction problems, the input and output elements may be pre-assigned to parts. The proposed hypergraph model can be simply accommodated to those problems by adding K *part* vertices and connecting those vertices to the nets which correspond to the pre-assigned input and output elements. Obviously, those part vertices must be fixed to corresponding parts during the partitioning. Since the required property is already included in the existing hypergraph partitioners [5, 13] this doesn't add extra complexity to our model. For the sparse matrix decomposition problem, we first assume that the consistency condition holds in the discussion throughout the following paragraphs and then discuss the appropriateness of the assumption in the last paragraph of this section.

Consider a partition $\Pi = \{\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_K\}$ of $\mathcal{H}$ in the fine-grain hypergraph model for 2D decomposition of matrix **A**. Without loss of generality, we assume that part $\mathcal{P}_k$ is assigned to processor P_k for $k = 1, 2, \ldots, K$. Recall that Π is defined as a partition on the vertex set of $\mathcal{H}$, hence it does not induce any part assignment for the nets. Since column and row nets of $\mathcal{H}$ denote the expand and fold operations on **x** and **y** vectors, we need to decode Π as inducing a partition on nets to formulate communication volume requirements. Let $\Lambda[n_j]$ and $\Lambda[m_j]$ denote the connectivity sets of column-net n_j and row-net m_j in Π, and $part[v_{ij}]$ denotes the part (hence processor) assignment for vertex v_{ij}.

Consider an internal column-net n_j of part $\mathcal{P}_k$ (i.e. $\Lambda[n_j] = \{\mathcal{P}_k\}$). As all pins of net n_j lie in $\mathcal{P}_k$, all nonzeros in column j (including a_{jj} by the consistency condition) which need x_j for their multiplication are already assigned to processor P_k. Hence, internal column-net n_j of $\mathcal{P}_k$, which does not contribute to the cutsize (3) of partition Π, does not necessitate any expand operation if x_j is assigned to processor P_k. Similarly, consider an internal row-net m_j of part $\mathcal{P}_k$. As all pins of row-net m_j lie in $\mathcal{P}_k$, all nonzeros in row j which will contribute to the accumulation of y_j are already assigned to processor P_k. Hence, internal row-net m_j of $\mathcal{P}_k$, which does not contribute to the cutsize (3) of partition Π, does not necessitate any fold operation if y_j is assigned to processor P_k.

Consider an external column-net n_j (i.e., $\lambda[n_j] > 1$). As all pins of net n_j lie in the parts in its connectivity set $\Lambda[n_j]$, all nonzeros (including a_{jj} by the consistency condition) which need x_j for multiplication are assigned to the parts (processors) in $\Lambda[n_j]$. Hence, contribution $\lambda[n_j] - 1$ of external net n_j to the cutsize according to (3) accurately models the amount of communication volume to incur during the expand of x_j if x_j is assigned to any processor in $\Lambda[n_j]$. Let $map[n_j] \in \Lambda[n_j]$ denote the part and hence processor assignment for x_j corresponding to cut net n_j. Cut net n_j indicates that processor $map[n_j]$ should send its local x_j to those processors in connectivity set $\Lambda[n_j]$ except itself (i.e., to processors in the set $\Lambda[n_j] - \{map[n_j]\}$). Hence, processor $map[n_j]$ should send its local x_j to $|\Lambda[n_j]| - 1 = \lambda[n_j] - 1$ distinct processors. Similarly, consider an external row-net m_j. As all pins of net m_j lie in the parts in its connectivity set $\Lambda[m_j]$, all nonzeros which will contribute in the accumulation of y_j are already assigned to the parts (processors) in $\Lambda[m_j]$. Cut net m_j indicates that the processors in the connectivity set $\Lambda[m_j]$ except owner of m_j (i.e., processors in the set $\Lambda[m_j] - \{map[m_j]\}$) should send their partial y_j results to the processor $map[m_j]$. Hence, contribution $\lambda[m_j] - 1$ of external row-net m_j to the cutsize according to (3) accurately models the amount of communication volume to incur during the fold of y_j if y_j is assigned to any processor in $\Lambda[m_j]$.

The above discussion shows that the consistency of the proposed hypergraph model depends on the existence of an at least one common part between connectivity sets $\Lambda[n_j]$ and $\Lambda[m_j]$, for each $j = 1, \ldots, M$. Fortunately, the consistency condition also guarantees the existence of such a common part which is $part[v_{jj}]$, because nets n_j and m_j share pin v_{jj} by the consistency condition. In essence, in the fine-grain hypergraph model, any partition Π of $\mathcal{H}$ with $part[v_{ii}] = \mathcal{P}_k$ can be safely decoded as assigning row-net m_i (hence y_i) and column-net n_i (hence x_i) to part $\mathcal{P}_k$, i.e., $map[n_i] = map[m_i] = part[v_{ii}]$. With this assignment, both symmetric partitioning on $\mathbf{x}$ and $\mathbf{y}$ vectors is maintained and also total communication volume is exactly modeled. Thus, in the fine-grain model, minimizing the cutsize according to (3) corresponds to minimizing the actual volume of interprocessor communication during the pre and post communication phases.

Nonzero diagonal entries automatically satisfy the condition "$v_{ii} \in pins[m_i]$ and $v_{ii} \in pins[n_i]$ for each row-net m_i and column-net n_i" thus enabling both accurate representation of communication requirement and symmetric partitioning of **x** and **y** vectors. If however some diagonal entries of the given matrix are zero then the consistency of the proposed model is easily maintained by simply adding dummy vertex v_{jj} for each $a_{jj} = 0$ with $w_{jj} = 0$ to vertex set $\mathcal{V}$ of $\mathcal{H}$. Vertex v_{jj} is also added to both pin lists $pins[n_j]$ and $pins[m_j]$. The net list of this dummy vertex v_{jj} is simply set to $nets[v_{jj}] = \{n_j, m_j\}$. Since dummy vertices are assigned zero weight, these vertex additions do not affect the balancing model (1).

4 Experimental Results

We have tested the validity of the proposed hypergraph model for 2D decomposition by running our multilevel hypergraph partitioning tool PaToH [5] on the hypergraphs for the decompositions of various realistic sparse test matrices arising in different application domains [1, 6, 7, 8]. Table 1 illustrates the properties of the test matrices listed in the order of increasing number of nonzeros. The proposed 2D decomposition results were compared with the 1D decompositions obtained by running MeTiS [12] using the standard graph models, and PaToH using the 1D column/row-net hypergraph model presented in [4]. All experiments were carried out on a workstation equipped with a 133 MHz PowerPC processor with 64 Mbytes of memory. For a specific K value, K-way decomposition of a test matrix constitutes a decomposition instance. MeTiS and PaToH were run 50 times starting from different random seeds for each decomposition instance and average performance results are displayed in Table 2. The percent load imbalance values are below 3% for all decomposition results displayed in these figures, where percent imbalance ratio is defined as $100 \times (W_{max} - W_{avg})/W_{avg}$.

Table 1. Properties of test matrices

name	number of rows/cols	number of nonzeros total	per row/col min	max	avg
sherman3	5005	20033	1	7	4.00
bcspwr10	5300	21842	2	14	4.12
ken-11	14694	82454	2	243	5.61
nl	7039	105089	1	361	14.93
ken-13	28632	161804	2	339	5.65
cq9	9278	221590	1	702	23.88
co9	10789	249205	1	707	23.10
pltexpA4-6	26894	269736	5	204	10.03
vibrobox	12328	342828	9	121	27.81
cre-d	8926	372266	1	845	41.71
cre-b	9648	398806	1	904	41.34
world	34506	582064	1	972	16.87
mod2	34774	604910	1	941	17.40
finan512	74752	615774	3	1449	8.24

Table 2 displays the decomposition performance of the proposed fine-grain hypergraph model for 2D decomposition together with the standard graph model and 1D hypergraph model. Communication volume values (in terms of the number of words transmitted) are scaled by the number of rows/columns of the respective test matrices. Although the main objective of this work is the minimization of the total communication volume, average number of messages handled by a single processor are also displayed in this table. As seen in Table 2, the proposed 2D hypergraph model produces substantially better partitions than 1D decomposition models at each instance in terms of total communication volume. On the overall average, 2D fine-grain hypergraph model produces 59%, and 43% better decompositions than 1D graph and hypergraph models, respectively.

Table 2 also displays the average number of messages handled by a single processor. Recall that, the theoretical bound on the maximum number of messages handled by a single processor is $K-1$ for the standard graph model and 1D hypergraph models, whereas it is $2(K-1)$ for fine-grain hypergraph model. Although, fine-grain hypergraph model produces 39% worse decompositions than the standard graph model in terms of average number of messages for $K = 16$, it only requires 16.17 messages per processor on the average which is well below the theoretical bound $2(16-1) = 30$. With increasing K, average number of messages becomes more comparable with the standard graph model, for example for $K = 64$ they are almost same.

The average execution times of the MeTiS and PaToH for the standard graph and hypergraph models are also displayed in Table 2. 2D fine-grain hypergraph model is approximately 2.4 and 7.3 times slower than the 1D hypergraph model and the standard graph model, respectively. This is an expected result, since 2D fine-grain hypergraph model contains 2 times more pins and nets than the 1D hypergraph model. Also number of vertices in the 2D fine-grain model is equal to the number of nonzeros in the matrix, whereas it is the number of rows/columns in 1D hypergraph model. Here, we should note that we have used PaToH without any modification and tuning. We are expecting substantial decrease in the run-time of PaToH for fine-grain hypergraph model with planned modifications.

5 Conclusion

A fine-grain computational hypergraph model was proposed for two-dimensional (2D) decomposition of sparse matrices. The proposed model reduces the 2D matrix decomposition problem to the well-known hypergraph partitioning problem so that partitioning objectives correspond to minimizing communication volume while maintaining load balance during repeated matrix-vector multiplication. The

Table 2. Average communication requirements of the proposed 2D hypergraph model and the existing 1D decomposition models.

		1D Decomposition								2D Decomposition			
		Standard Graph Model				Hypergraph Model				Fine-Grain Hypergraph Model			
		comm. vol.		avg		comm. vol.		avg		comm. vol.		avg	
name	K	tot	max	#msgs	time	tot	max	#msgs	time	tot	max	#msgs	time
sherman3	16	0.31	0.03	5.30	0.53	0.25	0.02	4.46	(1.77)	0.25	0.02	8.38	(3.03)
	32	0.46	0.02	6.48	0.61	0.37	0.02	5.81	(1.79)	0.36	0.02	10.07	(3.34)
	64	0.64	0.02	7.42	0.71	0.53	0.01	6.94	(1.71)	0.50	0.01	11.01	(3.39)
BCSPWR10	16	0.09	0.01	4.21	0.28	0.08	0.01	4.29	(3.62)	0.07	0.01	7.14	(7.28)
	32	0.15	0.01	4.79	0.34	0.13	0.01	4.65	(3.63)	0.12	0.01	7.49	(7.25)
	64	0.23	0.01	5.20	0.42	0.22	0.01	4.93	(3.34)	0.19	0.01	7.32	(6.86)
ken-11	16	0.93	0.08	13.99	1.77	0.60	0.05	12.91	(2.19)	0.14	0.02	10.79	(3.66)
	32	1.17	0.06	26.00	1.98	0.74	0.03	21.19	(2.39)	0.29	0.02	18.85	(4.09)
	64	1.45	0.04	40.48	2.35	0.93	0.02	32.22	(2.26)	0.48	0.02	28.23	(4.20)
nl	16	1.70	0.15	14.99	1.21	1.06	0.10	13.30	(3.09)	0.74	0.08	23.87	(7.07)
	32	2.25	0.10	27.88	1.43	1.49	0.07	20.39	(3.12)	1.05	0.07	35.98	(7.39)
	64	3.04	0.07	38.35	1.54	2.20	0.05	26.13	(3.34)	1.38	0.05	42.43	(8.03)
ken-13	16	0.94	0.08	14.77	3.84	0.55	0.04	13.87	(2.17)	0.08	0.01	9.39	(3.33)
	32	1.17	0.05	29.02	4.50	0.63	0.03	22.79	(2.18)	0.17	0.02	11.22	(3.64)
	64	1.40	0.03	50.81	4.78	0.79	0.02	35.93	(2.30)	0.39	0.02	20.51	(4.33)
cq9	16	1.70	0.17	14.88	2.12	0.99	0.12	12.62	(2.64)	0.50	0.08	18.03	(6.81)
	32	2.43	0.15	21.96	2.46	1.45	0.08	17.87	(2.61)	0.79	0.09	24.54	(6.96)
	64	3.73	0.12	32.27	2.80	2.33	0.06	22.67	(2.82)	1.22	0.07	30.72	(7.31)
co9	16	1.50	0.16	14.81	2.42	0.94	0.11	12.82	(2.72)	0.47	0.07	20.00	(6.63)
	32	2.07	0.12	19.62	2.84	1.36	0.08	17.55	(2.78)	0.74	0.07	26.84	(7.14)
	64	3.10	0.09	29.99	3.07	2.17	0.06	21.85	(2.99)	1.09	0.06	31.13	(8.01)
pltexpA4-6	16	0.34	0.03	10.05	3.22	0.30	0.03	10.11	(3.81)	0.20	0.02	14.78	(8.92)
	32	0.55	0.03	15.86	3.84	0.51	0.02	14.73	(4.13)	0.29	0.01	20.51	(9.61)
	64	0.98	0.03	20.48	4.32	0.86	0.02	17.35	(4.21)	0.51	0.01	21.40	(9.73)
vibrobox	16	1.24	0.11	12.84	2.77	1.06	0.08	10.14	(4.56)	0.79	0.07	23.27	(10.40)
	32	1.73	0.08	20.85	3.25	1.53	0.06	14.77	(4.65)	1.06	0.06	31.28	(10.90)
	64	2.28	0.05	28.85	3.49	2.08	0.05	19.58	(4.97)	1.43	0.05	35.38	(11.88)
cre-d	16	2.82	0.24	14.90	4.18	2.00	0.17	11.78	(2.34)	1.15	0.12	26.05	(7.49)
	32	4.12	0.19	28.59	4.80	2.90	0.14	19.49	(2.44)	1.77	0.11	41.37	(8.08)
	64	5.95	0.14	47.36	5.03	4.14	0.10	29.73	(2.72)	2.55	0.10	55.76	(9.05)
cre-b	16	2.62	0.23	14.78	4.41	2.02	0.18	12.13	(2.38)	1.01	0.11	25.91	(7.27)
	32	3.90	0.18	28.57	5.01	2.88	0.15	19.97	(2.42)	1.55	0.11	40.33	(7.96)
	64	5.73	0.14	46.42	5.42	4.08	0.12	29.98	(2.62)	2.26	0.10	52.72	(8.66)
world	16	0.59	0.05	11.78	5.76	0.54	0.06	6.09	(3.36)	0.23	0.05	16.57	(8.37)
	32	0.84	0.04	18.00	7.04	0.76	0.05	8.19	(3.34)	0.41	0.04	23.14	(9.00)
	64	1.19	0.03	20.58	8.16	1.06	0.04	11.58	(3.54)	0.62	0.04	27.42	(9.54)
mod2	16	0.57	0.05	10.95	5.85	0.52	0.06	5.59	(3.51)	0.24	0.05	13.02	(8.92)
	32	0.79	0.04	14.59	7.19	0.72	0.04	7.42	(3.32)	0.41	0.05	18.68	(9.20)
	64	1.14	0.03	17.84	7.96	1.02	0.04	10.51	(3.68)	0.62	0.04	24.44	(9.33)
finan512	16	0.20	0.03	4.35	7.84	0.16	0.03	3.48	(3.28)	0.07	0.02	9.24	(7.03)
	32	0.27	0.02	6.39	9.56	0.21	0.02	4.15	(3.30)	0.10	0.02	10.75	(7.04)
	64	0.38	0.01	8.80	11.17	0.31	0.01	5.37	(3.34)	0.20	0.02	14.90	(7.13)
Averages over K													
average	16	1.11	0.10	11.61	3.30	0.79	0.08	9.54	(2.96)	0.42	0.05	16.17	(6.87)
	32	1.56	0.08	19.19	3.92	1.12	0.06	14.21	(3.01)	0.65	0.05	22.93	(7.26)
	64	2.23	0.06	28.20	4.37	1.62	0.04	19.63	(3.13)	0.96	0.04	28.81	(7.68)
overall average		1.63	0.08	19.67	3.86	1.18	0.06	14.46	(3.03)	0.68	0.05	22.64	(7.27)

"tot" denotes the total communication volume, whereas "max" denotes the maximum communication volume handled by a single processor. "avg #msgs" denotes the average number of messages per processor. "time" denotes the execution time (in seconds) of MeTiS and PaToH for the graph model and hypergraph models. Numbers in the parentheses are the normalized execution times with respect to graph model using MeTiS.

performance of the proposed 2D decomposition model was tested against 1D decomposition through graph and hypergraph models on a wide range of realistic sparse matrices. The 2D decompositions achieved about 50 percent decrease in the communication volume requirement of a single parallel matrix-vector multiplication, on the average.

References

[1] W. J. Carolan, J. E. Hill, J. L. Kennington, S. Niemi, and S. J. Wichmann. An empirical evaluation of the korbx algorithms for military airlift applications. *Operations Research*, 38(2):240–248, 1990.

[2] U. V. Çatalyürek. *Hypergraph Models for Sparse Matrix Partitioning and Reordering*. PhD thesis, Bilkent University, Computer Engineering and Information Science, Nov 1999.

[3] U. V. Çatalyürek and C. Aykanat. Decomposing irregularly sparse matrices for parallel matrix-vector multiplications. *Lecture Notes in Computer Science*, 1117:75–86, 1996.

[4] U. V. Çatalyürek and C. Aykanat. Hypergraph-partitioning based decomposition for parallel sparse-matrix vector multiplication. *IEEE Transactions on Parallel and Distributed Systems*, 10(7):673–693, 1999.

[5] U. V. Çatalyürek and C. Aykanat. *PaToH: A Multilevel Hypergraph Partitioning Tool, Version 3.0*. Bilkent University, Department of Computer Engineering, Ankara, 06533 Turkey, 1999.

[6] I. O. Center. Linear programming problems. ftp://col.biz.uiowa.edu:pub/testprob/lp/gondzio.

[7] T. Davis. University of florida sparse matrix collection: http://www.cise.ufl.edu/ davis/sparse/. *NA Digest*, 92/96/97(42/28/23), 1994/1996/1997.

[8] I. S. Duff, R. Grimes, and J. Lewis. Sparse matrix test problems. *ACM Transactions on Mathematical Software*, 15(1):1–14, march 1989.

[9] B. Hendrickson. Graph partitioning and parallel solvers: has the emperor no clothes? *Lecture Notes in Computer Science*, 1457:218–225, 1998.

[10] B. Hendrickson and T. G. Kolda. Graph partitioning models for parallel computing. *Parallel Computing*, 26:1519–1534, 2000.

[11] B. Hendrickson, R. Leland, and S. Plimpton. An efficient parallel algorithm for matrix-vector multiplication. *Int. J. High Speed Computing*, 7(1):73–88, 1995.

[12] 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, Department of Comp. Sci. and Eng., Army HPC Research Center, Minneapolis, 1998.

[13] G. Karypis, V. Kumar, R. Aggarwal, and S. Shekhar. *hMeTiS A Hypergraph Partitioning Package Version 1.0.1*. University of Minnesota, Department of Comp. Sci. and Eng., Army HPC Research Center, Minneapolis, 1998.

[14] T. Lengauer. *Combinatorial Algorithms for Integrated Circuit Layout*. Willey–Teubner, Chichester, U.K., 199.

[15] J. G. Lewis and R. A. van de Geijn. Distributed memory matrix-vector multiplication and conjugate gradient algorithms. In *Proceedings of Supercomputing'93*, pages 15–19, Portland, OR, November 1993.