SAND2007-7148C

Proceedings in Applied Mathematics and Mechanics, 29 October 2007

A Nested Dissection Approach for Sparse Matrix Partitioning

Erik G. Boman*

Sandia National Laboratories, Scalable Algorithms Dept, P.O. Box 5800, Albuquerque, NM 87185-1318, USA.

We consider how to partition and distribute sparse matrices among processors to reduce communication cost in sparse matrix computations, in particular, sparse matrix-vector multiplication. We consider 2d distributions, where the distribution is not constrained to just rows or columns. We present a new model and an algorithm based on vertex separators and nested dissection. Preliminary numerical results for sparse matrices from real applications indicate the new method performs consistently better than traditional 1d partitioning and is often also better than current 2d methods.

1 Introduction: Sparse Matrix-Vector Multiplication

Sparse matrix-vector multiplication is a common kernel in many computations, e.g., iterative solvers for linear systems of equations and PageRank computation. An important combinatorial problem is how to distribute the matrix and the vectors among processors to minimize the communication cost in parallel. We assume distributed-memory computers, where communication is expensive.

Sparse matrix-vector multiplication $y = Ax$ is usually parallelized such that the processor that owns element a_{ij} computes the contribution $a_{ij}x_j$. This is a local operation if x_j, y_i and a_{ij} all reside on the same processor; otherwise communication is required. In general, the following four steps are performed [1, 3]:

1) **Expand:** Send entries x_j to processors with a nonzero a_{ij} for some i.
2) **Local multiply-add:** $y_i := y_i + a_{ij}x_j$
3) **Fold:** Send partial y values to relevant processors.
4) **Sum:** Sum up the partial y values.

The matrix and vector partitioning problem we address is: Given a sparse matrix A and an integer $k > 1$, compute a parallel distribution over k processors of the nonzeros of A and also for the input and output vectors such that all processors have approximately equally many nonzeros (load balance), and the communication volume in sparse matrix-vector multiply is minimized. This is the most general form, but here we restrict ourselves to symmetric matrices where the input and output vectors must have the same distribution.

Typically, matrices are partitioned in a 1d fashion, either by rows or by columns. This partitioning problem has been modeled as graph partitioning in the symmetric case, but hypergraph partitioning is known to more accurately model communication volume [1], and is therefore often preferred today. Recently, several 2d decompositions have been proposed [2, 3], where the communication volume is reduced further by giving up the simplicity of the 1d structure. The fine-grain method [2] is most general and thus of particular interest. We present a new algorithm for sparse matrix partitioning based on vertex separators in the graph and compare it to existing methods.

2 An Accurate Graph Model for the Symmetric Case

The classic (1d) graph model where cut edges represent communication is inaccurate, so we present a new and accurate model. In the symmetric case, we restrict our attention to symmetric partitioning schemes, where (i, j) and (j, i) are assigned to the same processor. Let $G(V, E)$ be the graph of the symmetric matrix. Now assign each vertex and each edge to a partition. A vertex incurs communication iff there are incident edges that belong to a different partition. The communication volume for such a vertex is two (corresponding to the expand and fold phases) times the number of partitions involved (besides the current).

Theorem 2.1 *Let $G(V, E)$ be the graph of a symmetric matrix. Let $E(v)$ denote the set of edges incident to vertex v. Let $\pi(v)$ and $\pi(e)$ denote the partitions to which v and e belong, respectively. Then the communication volume in sparse matrix-vector multiplication is* $2\sum_{v\in V}(|\pi(v) \cup \pi(E(v))| - 1)$.

3 A Vertex Separator and Nested Dissection Algorithm

We first consider bisection, and later generalize to k-way partitioning for $k > 2$ using recursive bisection. Our idea is to use a vertex separator to bisect the graph of the matrix, and assign each half to a different partition (processor). All communication has to go via the separator, so a small separator limits the communication.

* Corresponding author: e-mail: egboman@sandia.gov,

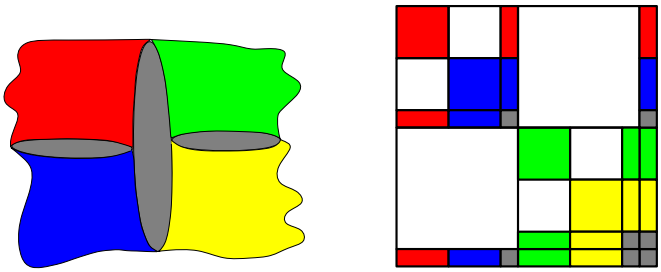

Fig. 1 Generic graph partitioned using nested dissection (left). The gray areas are the separators. The reordered and partitioned matrix is shown (right). The gray areas represent the separators where we have some flexibility in the assignment.

Our algorithm works as follows. First compute a small balanced vertex separator S for the graph using any vertex separator algorithm. This partitions the vertices into three disjoint subsets (V_0, V_1, S). Let $E_j := \{e \in E | e \cap V_j \neq \emptyset\}$ for $j = 0, 1$, that is, E_j is the set of edges with at least one endpoint in V_j. Assign V_j and E_j to partition (processor) j for $j = 0, 1$.

The procedure above intentionally does not specify how to distribute the vertices in S and the edges therein. There are several ways to exploit this flexibility, giving several variations. In the basic version, we simply assign all of S to the same partition, preferably the partition with the least data (edges or nonzeros).

For $k > 2$ parts, we apply the vertex separator bisection method recursively. This yields a recursive bisection algorithm very similar to the well-known *nested dissection* method for sparse matrix ordering. The case $k = 4$ is illustrated in Figure 3.

4 Results and Conclusions

We compare the communication volume for parallel sparse matrix-vector multiplication for a set of sparse matrices from different application areas (structural analysis, information retrieval, linear programming, circuit simulation, chemical engineering) that were used in [3]. We only use symmetric matrices here, and leave the nonsymmetric case to a future paper.

We compare a variation of our nested dissection algorithm to 1d hypergraph partitioning, Mondriaan, and fine-grain hypergraph partitioning. We used PaToH 3.0 as our hypergraph partitioner and Mondriaan 1.02. METIS is a good code for nested dissection ordering, but it does not directly provide the separators. Instead we derived vertex separators from the hypergraph partitioning, which also facilitates a fair, direct comparison. Due to randomization, we ran all tests 20 times and present the average. We see from Table 1 that no method is consistently best. Mondriaan won three times, fine-grain hypergraph five and nested dissection seven.

Table 1 Communication volume for k-way partitioning of symmetric test matrices using four different methods.

Name	k	1d hyp.	Mondriaan	fine-grain	nested diss.
cage10	4	5379.0	5051.0	**4063.7**	4418.7
	16	12874.5	11264.4	**8865.5**	11404.8
	64	23463.3	19266.0	**16334.7**	22963.5
hyp200_2	4	1535.1	**1464.5**	1538.5	1530.8
	16	3013.9	**2735.6**	3017.9	3012.9
	64	5813.0	**5030.4**	5786.4	5824.9
finan512	4	295.7	1099.8	261.2	**229.1**
	16	1216.7	1786.5	1027.4	**936.9**
	64	9986.0	9316.5	8624.6	**8060.7**

Name	k	1d hyp.	Mondriaan	fine-grain	nested diss.
bcsstk30	4	1794.4	1817.6	1935.7	**1401.3**
	16	8624.7	8527.9	9774.8	**7100.9**
	64	23308.0	23322.5	25677.2	**20814.1**
bcsstk32	4	2111.9	2343.1	**1611.4**	1773.1
	16	7893.1	8241.2	**6330.8**	6780.1
	64	19905.4	19495.5	18673.1	**18006.6**

We conclude our algorithm reduces communication volume compared to traditional 1d partitioning methods and is competitive with 2d methods. Though initially limited to symmetric matrices, it is possible to extend the nested dissection method to the nonsymmetric case.

Acknowledgements We thank Michael Wolf for providing these numerical results. This work was funded by the U.S. DOE Office of Science through SciDAC and the CSCAPES Institute. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed-Martin Company, for the DOE's National Nuclear Security Administration under contract number DE-AC-94AL85000.

References

[1] U.V. Catalyurek and C. Aykanat, Hypergraph-Partitioning Based Decomposition for Parallel Sparse-Matrix Vector Multiplication, IEEE TPDS, v. 10, no. 7, pp. 673–693, 1999.
[2] U.V. Catalyurek and C. Aykanat, A Fine-Grain Hypergraph Model for 2D Decomposition of Sparse Matrices, Proc. IPDPS, 2001.
[3] B. Vastenhouw and R.H. Bisseling, A two-dimensional data distribution method for parallel sparse matrix-vector multiplication, SIAM Rev., v. 47, no. 1, pp. 67–95, 2005.