# PARALLEL HYPERGRAPH PARTITIONING FOR SCIENTIFIC COMPUTING

E. BOMAN*, R. BISSELING , U. CATALYUREK , K. DEVINE , R. HEAPHY , AND B. HENDRICKSON

**1. Introduction.** Partitioning and load-balancing are important issues in parallel scientific computing. The goal is to distribute data (and work) evenly among processors such as to reduce communication cost and achieve maximal performance. Graph partitioning has long served as a useful model for load balancing in parallel computing. Data are represented as vertices in a graph, and edges represent communication requirements. Graph partitioning attempts to minimize the number of cross-edges in the graph between processors. It has been shown that for many problems, this is not an accurate representation of communication cost or volume. On the other hand, *hypergraph partitioning* accurately minimizes communication volume [1]. A hypergraph $H = (V, E)$ consists of a vertex set $V$ and a set of hyperedges $E$. Each hyperedge is a subset of $V$. In parallel computing, an hyperedge represents communication between two or more processors. Aykanat and Catalyurek [1] proposed a hypergraph model for sparse matrix-vector multiplication, and showed that the hyperedge cut corresponds *exactly* to the communication volume. An important advantage of the hypergraph model is that it can easily represent nonsymmetric and rectangular matrices.

An important kernel in many scientific computations is a sparse matrix-vector product. The parallel issue is how to distribute the sparse matrix between processors. The most common approach is to split the matrix either by rows or columns, and distribute approximately even chunks (of rows/columns) to processors. Both variations naturally lead to hypergraph partitioning. In the row-net model, each column corresponds to a vertex and each row corresponds to an hyperedge. For row partitioning, an analogous column-net model can be used. Although simply using hypergraph partitioning already gives an improvement over graph partitioning (25-35% reduction is typical [1, 6]), even lower communication volumes can be achieved by going beyond 1-d partitioning. Vastenhouw and Bisseling recently suggested a recursive 2-dimensional data distribution known as Mondriaan [6]. Catalyurek has proposed a fine-grain partitioning model [2] where each nonzero in a matrix is independently assigned to processors. Both these methods rely on hypergraph partitioning as an underlying technique. Software for hypergraph partitioning therefore becomes important.

**2. The new parallel hypergraph partitioner in Zoltan.** Several software packages for hypergraph partitioning exists; e.g., Patoh [**?**], HMetis [4], and Mondriaan [6] (for sparse matrices). However, all these packages run in serial. For large-scale parallel applications it is necessary to perform the partitioning itself in parallel. The Zoltan [3] library was developed at Sandia as a toolkit for parallel partitioning and load-balancing. In the following, we describe the design and structure of a parallel hypergraph partitioner under development for Zoltan. Concurrently with this project, Trifunovic and Knottenbelt have also developed a parallel hypergraph partitioner, Parkway [5].

*Corresponding author. egboman@sandia.gov

Our hypergraph partitioner is based on the well established multilevel partitioning method. In a *coarsening* phase the hypergraph is reduced in size by merging vertices until the graph is small enough to be partitioned by a simple greedy method. Our coarsening is based on greedy maxinmal weight matching, where the edge weights are computed as inner products between vertices (heavy connectivity matching). In the *uncoarsening* phase, the resulting partitions are propagated back to the initial hypergraph. Local refinement (KL/FM) is applied on every level to improve quality. Both coarsening and uncoarsening are computed in parallel in a distributed way, while the coarse (greedy) partitioning is computed independently by each processor in a trivial parallel fashion.

Our implementation is novel in that it uses a 2-dimensional data distribution. The hypergraph is represented as a sparse matrix, where rows are hyperedges. This sparse matrix is partitioned both along rows and columns, and each resulting submatrix (block) is assigned to a different processor. Conceptually the processors are organized as a regular 2-d grid. An advantage of this design is that most communication is restricted to either rows or columns of processors. We do not use any "ghosting" of off-processor data, so our implementation is memory efficient and scalable.

The Zoltan hypergraph partitioner allows for arbitrary number of partitions and processors. When $k > 2$ partitions are requested, recursive bisection is used. For the subproblems, we have two options: Either leave the data in place such that all processors participate in all subproblems, or move the data between processors such that subsets of processors can work on independent subproblems. The latter approach incurs a data redistribution cost but saves communication within the partitioner. We study this trade-off.

Preliminary results show that our parallel partitioner is comparable to Patoh in both run-time and quality when run on a single processor. With increasing number of processors, the run-time decreases while the partitioning quality remains approximately constant. Scalability is currently good for up to 32-64 processors, and the design allows for scalability to thousands of processors. Empirical experiments on large sparse matrices (hypergraphs) will be presented in our talk.

REFERENCES


[1] Ü. Çatalyürek and C. Aykanat. Hypergraph-partitioning based decomposition for parallel sparse-matrix vector multiplication. *IEEE Trans. Parallel Dist. Systems*, 10(7):673–693, 1999.

[2] Ü. Çatalyürek and C. Aykanat. A fine-grain hypergraph model for 2d decomposition of sparse matrices. In *Proc. IPDPS 8th Int'l Workshop on Solving Irregularly Structured Problems in Parallel (Irregular 2001)*, April 2001.

[3] Karen Devine, Erik Boman, Robert Heaphy, Bruce Hendrickson, and Courtenay Vaughan. Zoltan data management services for parallel dynamic applications. *Computing in Science and Engineering*, 4(2):90–97, 2002.

[4] George Karypis, Rajat Aggarwal, Vipin Kumar, and Shashi Shekhar. Multilevel hypergraph partitioning: application in VLSI domain. In *Proc. 34th Conf. Design Automation*, pages 526 – 529. ACM, 1997.

[5] Aleksandar Trifunovic and William J. Knottenbelt. Parkway 2.0: A parallel multilevel hypergraph partitioning tool. In *Proc. 19th International Symposium on Computer and Information Sciences (ISCIS 2004)*, volume 3280 of *LNCS*, pages 789–800. Springer, 2004.

[6] Brendan Vastenhouw and Rob H. Bisseling. A two-dimensional data distribution method for parallel sparse matrix-vector multiplication. Preprint 1238, Dept. Mathematics, Utrecht University, May 2002. To appear in *SIAM Review*, 2005.