SAND2015-8747R

Advances in Domain Mapping of Massively Parallel Scientific Computations

Bruce Hendrickson and Robert Leland
Sandia National Laboratories
Albuquerque, NM 87185

One of the most important concerns in parallel computing is the proper distribution of workload across processors. For most scientific applications on massively parallel machines, the best approach to this distribution is to employ *data parallelism*; that is, to break the datastructures supporting a computation into pieces and then to assign those pieces to different processors. Collectively, these partitioning and assignment tasks comprise the *domain mapping* problem.

Extensive practical experience has shown that in order to achieve high performance, a domain mapping must be found in which each processor has about the same workload and in which the overhead due to interprocessor communication is kept small. Unfortunately, discovering such a mapping can be difficult, particularly for the large, unstructured domains typical of modern scientific/engineering calculations.

Many approaches to this problem have been tried over the years, but the results have often been disappointing. Approximate global optimization strategies such as simulated annealing and genetic algorithms are capable of finding excellent mappings, but are too expensive in practice.Simple heuristics based on ordering of coordinate or topological information are quick but they perform poorly on many complicated grids. More sophisticated heuristics typically show erratic behavior or are very problem specific.

We have recently developed several effective and economical domain mapping methods that are appropriate for finite difference, finite element, particle-in-cell and similar types of scientific computations. Our methods employ a graph model of the calculation in which vertices represent computation and edges represent communication. This model reduces the decomposition problem to one of graph partitioning, i.e. dividing the graph into sets with equal numbers of vertices such that a minimal number of edges cross between sets (an NP-hard problem). The assignment problem then requires that we find a mapping of these sets to processors that avoids messages between architecturally distant processors.

One of our new methods is a generalization of a recently proposed graph partitioning approach which has generated considerable interest because it seems to offer a good balance between generality, quality and efficiency. This *spectral* method partitions a graph by considering an eigenvector of an associated matrix to gain an understanding of global properties of the graph [9,10]. The graph is bisected using this eigenvector and the process repeated recursively on each of the halves until the desired number of sets is obtained.

We have improved this spectral technique in several important ways. First, we have made realistic modeling of scientific computations possible by allowing for unbalanced computation and communication loads in our graph model. Second, we have reformulated the objective

function so that we try to minimize not only the amount of communication between processors, but also the number of messages traveling between distant processors in a hypercube or mesh architecture. We have therefore addressed, for these important architectures, both the decomposition and assignment problems. (Most previous methods, including the original spectral method, have considered only the decomposition problem.) Third, we have developed a method of using two or three eigenvectors to partition the graph into four or eight pieces at each stage of recursion rather than simply bisecting based on one eigenvector. This results in better mappings at a lower net cost than that incurred by other spectral methods. Fourth, we have paired the spectral method with a generalization of a well known graph partitioning technique due to Kernighan and Lin (KL) [7]. The KL algorithm is a local optimization which improves an initial partitioning. The spectral method is quite good at coarsely dividing a graph, but often does poorly in the fine details, whereas KL exhibits precisely the opposite behavior. When paired, the combined method is substantially more robust and effective than either alone. This work has been reported in several publications [2,3,4,5,6].

While spectral methods are much more economical than simulated annealing and other methods capable of finding very high quality partitions, they are still costly in comparison with the actual run time of many applications. There is consequently a strong interest in developing comparably powerful mapping methods with substantially lower cost. One approach is to accelerate the eigenvector computation in the spectral method by employing a multi-level iterative eigensolver. The graph is coarsened down several levels, an eigenvector is computed on the coarsest grid where it is relatively cheap to do so, and the eigenvector is then projected back up through the grid hierarchy to the finest grid, undergoing a refinement process as it is projected. This method was proposed and implemented by Barnard and Simon [1], who found that it did substantially reduce the run time of spectral bisection.

We have developed an alternative approach in which rather than propagating an eigenvector of thecoarsest grid back through the heirarchy, we instead propagate a partitioning. We use a spectral method to determine the partitioning of the coarsest grid, and refine the partition as it is projected onto finer grids using the Kernighan-Lin algorithm described earlier. This approach avoids various numerical difficulties associated with the computation and refinement of eigenvectors of very large matrices, and typically produces comparable partitions in less time than those obtained with multilevel spectral bisection. Furthermore, this new multilevel method retains the ability to operate as a quadrisection or octasection algorithm, and can be applied to weighted graphs transparently.

We have tested these new methods on a number of unstructured grids from finite element, finite difference and particle-in-cell application codes. We have selected two representative problems for presentation here. The first is the dual graph of a finite element airfoil mesh generated by Barth at NASA Ames. This graph has 8034 vertices and 11813 edges, and is the subject of the three figures. The decomposition into eight sets depicted in Figure 1 was generated using the popular *inertial* method [8,11]. Figure 2 shows a decomposition using the original spectral bisection algorithm, while Figure 3 presents a decomposition produced using our new spectral octasection + KL method. In each case elements sharing a color are assigned to the same processor.

It is clearly difficult to judge the merits of these partitions visually, even for this relatively small two-dimensional problem. This reflects the general difficulty of the domain mapping problem. We can however calculate objective measures of success with respect to our graph model, and these are given in Table 1 for a variety of different partitioning methods. The *cuts* values are the number of edges that connect vertices assigned to different sets, which corresponds to the total volume of interprocessor communication. The values in the *hops* columns weight each cut edge by the number of wires between the corresponding processors in a hypercube network. In most scientific applications, many messages are being simultaneously routed, and this second metric accounts for message congestion in a mesh or hypercube multiprocessor.

	8 Processors		64 Processors	
Method	cuts	hops	cuts	hops
Inertial	317	396	1166	1855
Spectral Bisection	212	286	997	1661
Spectral Bisection + KL	190	261	871	1472
Spectral Octasection + KL	197	200	911	1287
Multi-Level Bisection	197	276	855	1467
Multi-Level Octasection	240	240	991	1414

Table 1: Summary of method performance on airfoil mesh.

The second example mesh is a three-dimensional finite element mesh of a complex manufacturing component generated using advanced meshing software at Sandia National Laboratories. This graph has 6673 vertices and 55664 edges. Results of a partition into eight sets are given in Table 2, and the decomposed mesh is shown in Figure 4.

	Sandia Mesh	
Method	cuts	hops
Inertial	4652	5594
Spectral Bisection	3425	5084
Spectral Bisection + KL	2562	3847
Spectral Octasection	4138	4735
Spectral Octasection + KL	3140	3420
Multi-Level Bisection	2577	3780
Multi-Level Octasection	3365	3514

Table 2: Performance of partitioning algorithms on machine part mesh.

For both these problems spectral octasection plus KL was clearly the best method at minimizing hops. For minimizing cuts, the multi-level bisection algorithm and spectral bisection plus KL were about equivalent, but the multilevel algorithm ran significantly faster. The multi-level octasection method, while still competitive, did not perform as well as expected. We believe this

is because a simple local strategy like Kernighan-Lin has difficulty refining partitions into more than two sets.

The code that produced these results is available for public use; requests should be directedto the authors.

References

[1] S. BARNARD AND H. SIMON, A fast multilevel implementation of recursive spectral bisection for partitioning of unstructured problems, in Proc. Parallel Processing for Scientific Computing, SIAM, 1993.

[2] B. HENDRICKSON AND R. LELAND, Domain mapping for massively parallel scientific computation, tech. rep., Sandia National Laboratories, 1992. MPCRL Bulletin, vol.2, no. 2.

[3] ____, An improved spectral graph partitioning algorith for mapping parallel computations, tech. rep., Sandia National Laboratories, 1992. SAND92-1460.

[4] ____, Multidimensional spectral load balancing, tech. rep., Sandia National Laboratories, 1992. SAND93-0074.

[5] ____, An improved spectral load balancing method, in Proc. Parallel Processing for Scientific Computing, SIAM, 1993.

[6] ____, A multi-level algorithm for partitioning graphs, tech. rep., Sandia National Laboratories, 1993. SAND93-1301.

[7] B. KERNIGHAN AND S. LIN, An efficient heuristic procedure for partitioning graphs, Bell System Tehcnical Journal, 29 (1970), pp. 291-307.

[8] B. NOUR-OMID, A. RAEFSKY, AND G. LYZENGA, Solving finite element equations on concurrent computers, in Parallel computations and their impact on mechanics, American Soc. of Mech. Eng., 1986. ed. A.K. Noor, pp. 209-227.

[9] A. POTHEN, H. SIMON, AND K. LIOU, Partitioning sparse matrices with eigenvectors of graphs, SIAM J. Matrix Anal, 11 (1990), pp. 430-452.

[10] H. SIMON, Partitioning of unstructured problems for parallel processing, in Proc. Conference on Parallel Methods on Large Scale Structural Analysis and Physics Applications, Pergammon Press, 1991.

[11] R. WILLIAMS, Performanceof dynamic load balancing algorithms for unstructured mesh calculations, J. Par. Dist. Comput., 13 (1991), pp. 257-272.

*This work was supported by the Applied Mathematical Sciences program, U.S. Department of Energy, Office of Energy Research, and was performed at Sandia National Laboratories, operated for the U.S. Department of Energy under contract No. DE-AC04-76DP00789.