A Domain Decomposition Solver for a Parallel Adaptive Meshing Paradigm

Randolph E. Bank *

Department of Mathematics, University of California at San Diego, La Jolla, California 92093-0112, USA. rbank@ucsd.edu

Summary. We describe a domain decomposition algorithm for use in the parallel adaptive meshing paradigm of Bank and Holst. Our algorithm has low communication, makes extensive use of existing sequential solvers, and exploits in several important ways data generated as part of the adaptive meshing paradigm. Numerical examples illustrate the effectiveness of the procedure.

1 Bank-Holst Algorithm

In [4, 3], we introduced a general approach to parallel adaptive meshing for systems of elliptic partial differential equations. This approach was motivated by the desire to keep communications costs low, and to allow sequential adaptive software (such as the software package PLTMG used in this work) to be employed without extensive recoding. Our discussion is framed in terms of continuous piecewise linear triangular finite element approximations used in PLTMG, although most ideas generalize to other approximation schemes. 

Our original paradigm, called Plan A in this work, has three main components:

Step I: Load Balancing. We solve a small problem on a coarse mesh, and use a posteriori error estimates to partition the mesh. Each subregion has approximately the same error, although subregions may vary considerably in terms of numbers of elements or gridpoints.

Step II: Adaptive Meshing. Each processor is provided the complete coarse mesh and instructed to sequentially solve the entire problem, with the stipulation that its adaptive refinement should be limited largely to

*The work of this author was supported by the National Science Foundation under contract DMS-0208449. The UCSD Scicomp Beowulf cluster was built using funds provided by the National Science Foundation through SCREMS Grant 0112413, with matching funds from the University of California at San Diego.
its own partition. The target number of elements and grid points for each problem is the same. At the end of this step, the mesh is regularized such that the global mesh described in Step III is conforming.

**Step III: Global Solve.** The final global mesh consists of the union of the refined partitions provided by each processor. A final solution is computed using domain decomposition.

With this paradigm, the load balancing problem is reduced to the numerical solution of a small elliptic problem on a single processor, using a sequential adaptive solver such as PLTMG without requiring any modifications to the sequential solver. The bulk of the calculation in the adaptive meshing step also takes place independently on each processor and can also be performed with a sequential solver with no modifications necessary for communication. The only parts of the calculation requiring communication are (1) the initial fan-out of the mesh distribution to the processors at the beginning of the adaptive meshing step, once the decomposition is determined by the error estimator in load balancing; (2) the mesh regularization, requiring communication to produce a global conforming mesh in preparation for the final global solve in Step III; and (3) the final solution phase, that requires communicating certain information about the interface system (see Section 2).

In [2], we considered a variant of the above approach in which the load balancing occurs on a much finer mesh. The motivation was to address some possible problems arising from the use of a coarse grid in computing the load balance. In particular, we assume in Plan A that $N_c \gg p$ where $N_c$ is the size of the coarse mesh and $p$ is the number of processors. This is necessary to allow the load balance to do an adequate job of partitioning the domain into regions with approximately equal error. We also assume that $N_c$ is sufficiently large and the mesh sufficiently well adapted for the a posteriori error estimates to accurately reflect the true behavior of the error. For the second step of the paradigm, we assume that $N_p \gg N_c$ where $N_p$ is the target size for the adaptive mesh produced in Step II of the paradigm. Taking $N_p \gg N_c$ is important to marginalize the cost of redundant computations.

If any of these assumptions is weakened or violated, there might be a corresponding decline the effectiveness of the paradigm. In this case, we consider the possibility of modifying Steps I and II of the paradigm as follows. This variant is called *Plan B* in this work.

**Step I: Load Balancing.** On a single processor we adaptively create a *fine* mesh of size $N_p$, and use a posteriori error estimates to partition the mesh such that each subregion has approximately equal error, similar to Step I of the original paradigm.

**Step II: Adaptive Meshing.** Each processor is provided the complete adaptive mesh and instructed to sequentially solve the *entire* problem. However, in this case each processor should adaptively *coarsen* regions corresponding to other processors, and adaptively refine its own subregion. The size of the problem on each processor remains $N_p$, but this adaptive