

# Performance of Fully-Coupled Algebraic Multilevel Domain Decomposition Preconditioners for Incompressible Flow and Transport

Paul T. Lin<sup>1\*</sup>, Marzio Sala<sup>2</sup>, John N. Shadid<sup>1</sup>, Ray S. Tuminaro<sup>3</sup>

<sup>1</sup> Sandia National Laboratories, PO Box 5800 MS 0316, Albuquerque, NM 87185-0316, U.S.A. <sup>2</sup> Sandia National Laboratories, PO Box 5800 MS 1110, Albuquerque, NM 87185-1110, U.S.A. <sup>3</sup> Sandia National Laboratories, PO Box 969 MS 9159, Livermore, CA 94551-9159, U.S.A.

## SUMMARY

This study investigates algebraic multilevel domain decomposition preconditioners of the Schwarz type for solving linear systems associated with Newton-Krylov methods. The key component of the preconditioner is a coarse approximation based on algebraic multigrid ideas to approximate the global behavior of the linear system. The algebraic multilevel preconditioner is based on an aggressive coarsening graph partitioning of the non-zero block structure of the Jacobian matrix. The scalability of the preconditioner is presented as well as comparisons with a two-level Schwarz preconditioner using a geometric coarse grid operator. These comparisons are obtained on large-scale distributed-memory parallel machines for systems arising from incompressible flow and transport using a stabilized finite element formulation. The results demonstrate the influence of the smoothers and coarse level solvers for a set of 3D example problems. For preconditioners with more than one level, careful attention needs to be given to the balance of robustness and convergence rate for the smoothers and the cost of applying these methods. For properly chosen parameters, the two- and three-level preconditioners are demonstrated to be scalable to 1024 processors. Copyright © 192004 John Wiley & Sons, Ltd.

KEY WORDS: multilevel preconditioners, multigrid, finite element methods, Newton-Krylov, Schwarz domain decomposition preconditioners, graph partitioning

## 1. INTRODUCTION

Computational fluid dynamic simulations often require the solution of strongly-coupled interacting physics on high resolution unstructured meshes. The discretization and linearization of the equations produce large linear systems of equations, for which robust and efficient parallel iterative solution methods are necessary. Preconditioned Krylov iterative methods are among the most robust and fastest iterative solvers over a wide variety of CFD applications [1]. The key factor influencing the robustness

---

\*Correspondence to: Sandia National Laboratories, P.O. Box 5800 MS 0316, Albuquerque, NM 87185-0316, U.S.A.

Contract/grant sponsor: ASCI program and the DOE Office of Science MICS program at Sandia National Laboratory. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

and efficiency of these solution methods is the choice of preconditioners. Multilevel Schwarz domain decomposition based preconditioners [2] will be considered in this study. The additive Schwarz approach partitions the original domain into overlapping subdomains and approximately solves the discrete problem corresponding to the individual subdomains in parallel. A known disadvantage is that this method (referred to as a one-level Schwarz preconditioner) does not scale as the number of subdomains increases. That is, the number of iterations rises as the number of subdomains increases. To remedy this situation, multigrid ideas can be employed. The general multigrid philosophy is to use a sequence of meshes to damp errors at different frequencies thereby accelerating the convergence to the solution on the finest mesh. In a typical multigrid method, many coarse meshes are used in conjunction with fairly lightweight smoothers. The coarsening rate to define the next mesh is normally fairly modest (e.g. the next coarser mesh contains one tenth as many grid points as the previous finer mesh). In a domain decomposition setting, only one coarse mesh is usually employed and this mesh is significantly coarser than the finest mesh. The one-level Schwarz preconditioner can be viewed as a heavyweight somewhat expensive smoother from a multigrid perspective. This expensive smoother is natural when the coarse grid is much coarser than the fine mesh.

Multigrid methods come in two varieties. Geometric multigrid methods use grid coordinate type information (and perhaps finite element information) to define grid transfers. While these methods can work well, they typically require users to define a sequence of meshes along with operators to transfer solutions between them. The other variety of multigrid fall into the category of algebraic methods. In this case, the coarse meshes as well as the grid transfers are defined using only information from the fine grid linear system. While the generation of these operators is reasonably understood for Poisson-like operators [3], optimal grid transfers for highly convective flows and chemical reactions is still not well-understood in the context of algebraic methods.

In this paper we present a scalable multilevel preconditioner based on aggregation [4, 5]. This work is an extension of previous work [6] where a two-level preconditioner with geometric coarse operator was compared with the one-level counterpart. Here, we extend this analysis by comparing several multilevel domain decomposition preconditioners for accelerating the convergence of a parallel Newton-Krylov solution method. Aggregation has been used to define two-level domain decomposition preconditioners by several authors [7, 8, 9]. One important aspect is to evaluate whether a two-level preconditioner with an algebraic coarse operator could perform (in terms of iterations and parallel scalability) as well as a two-level preconditioner with a geometric coarse operator. In [10], some results are reported that compare two-level preconditioners with geometric coarse grids with smoothed and nonsmoothed aggregation techniques for a Laplacian problem, showing that both methods share comparable behavior for symmetric coercive problems. Here, we numerically compare the two approaches for nonsymmetric problems, namely the incompressible Navier-Stokes equations with heat transfer and convection-diffusion equation with a given velocity field. We show that, even for problems defined on simple geometries, the algebraic preconditioner is not worse than the geometric one. To the best of our knowledge, these results are the first presented in the literature that compare two-level preconditioners using algebraic coarse operators with those using geometric coarse operators on massively parallel computers. We also present a scalable three-level algebraic method, and we show that we can drastically reduce the CPU-time to solve fluid flow problems by using three-level aggregation-based preconditioners.

## 2. Governing Equations

The governing PDEs describing fluid flow, thermal energy transfer, and mass transfer for variable density low-speed flow are as follows.

Total Mass Conservation:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0$$

Momentum Transport:

$$\rho \frac{\partial \mathbf{u}}{\partial t} + \rho(\mathbf{u} \cdot \nabla \mathbf{u}) - \nabla \cdot \mathbf{T} - \rho \mathbf{g} = 0$$

Energy Transport:

$$\rho C_p \left[ \frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T \right] - \nabla \cdot (\lambda \nabla T) = 0$$

Species Transport:

$$\rho \left[ \frac{\partial Y_k}{\partial t} + \mathbf{u} \cdot \nabla Y_k \right] - \nabla \cdot (\rho D_k \nabla Y_k) = 0 \quad k = 1, 2, \dots, N_s - 1$$

Constitutive equation for Newtonian stress tensor:

$$\mathbf{T} = -P\mathbf{I} - \frac{2}{3}\mu(\nabla \cdot \mathbf{u})\mathbf{I} + \mu[\nabla \mathbf{u} + \nabla \mathbf{u}^T]$$

In these equations the unknowns are the velocity vector  $\mathbf{u}$ , the temperature  $T$ , the hydrodynamic pressure  $P$ , and the  $N_s$  species mass fractions  $Y_k$ . The transport properties,  $\rho, \mu, C_p, \lambda, D_k$ , are respectively, the density, dynamic viscosity, specific heat capacity, heat conductivity, and diffusion coefficient. These equations are approximated by a stabilized finite element formulation [11, 12, 13] that allows for equal order interpolation of pressure and velocity (without spurious pressure solutions) and for stabilization of highly convected flows.

The computer code used in this study is MPSalsa [13], a parallel finite element method on unstructured meshes that solves incompressible reacting flow problems.

## 3. Preconditioned Newton-Krylov Method

The discretized equations are solved using a Newton-Krylov method [14, 15], which is an implementation of Newton's method in which a Krylov accelerator is used to solve the linear systems that are generated at each step of Newton's method. A Newton-Krylov method is usually implemented as an inexact Newton method [16], meaning that the linear systems are solved only approximately.

For the considered class of problems, convergence cannot be achieved without preconditioning [17]. One widely used preconditioner, well-suited for parallel computations, is the one-level Schwarz preconditioner [2, 18]. The procedure is as follows: first, we decompose the computational domain  $\Omega$  into  $M$  overlapping subdomains  $\Omega_i$ , and we assign each subdomain to a different processor. Then, the preconditioner is applied by solving, usually with a direct method, a Dirichlet problem on each subdomain  $\Omega_i$  (with homogeneous boundary conditions on  $\partial\Omega_i$ ). Using a minimal overlap, the resulting preconditioner can be seen as a block Jacobi preconditioner, where each block contains all the nodes assigned to a given subdomain. Better performance can be obtained by increasing the overlap between the subdomains.

Because of the large cost of direct factorization techniques, an approximate factorization technique is employed for the solution of the local Dirichlet problems. For this work, ILUT [19] is used, typically with a drop tolerance of zero and keeping the number of nonzeros in the ILUT factors approximately equal to the number of nonzeros in the original discretization matrix. The Aztec library [20] is used to implement the Krylov accelerator and the one-level Schwarz preconditioner. We note that the one-level preconditioner is completely black-box, since Aztec automatically constructs the overlapping submatrices. However, a drawback of this preconditioner is that as the number of subdomains increases, the convergence rate deteriorates due to the lack of global coupling in the preconditioner [2]. To remedy this situation, coarse operators can be added to approximate the global coupling in the linear system [21, 6].

#### 4. Multilevel Preconditioners

Our implementation of the presented multilevel preconditioners is based on the ML multilevel preconditioning package [21, 22]. For the two-level Schwarz preconditioner with a geometric coarse operator, ML expects the user to supply both the fine and coarse meshes as well as finite element basis functions on the coarse mesh. Using this information, ML constructs the interpolation or prolongation operator that corresponds to the coarse mesh basis functions. ML handles all of the bookkeeping (such as determining the coarse mesh element that contains each fine grid point) and uses callback functions to the application providing a data-structure neutral interface. The restriction operator is calculated as the transpose of the projection operator. The coarse matrix  $\mathbf{A}_c$  is defined by the Galerkin projection  $\mathbf{A}_c = \mathbf{R}\mathbf{A}_f\mathbf{P}$ , where  $\mathbf{R} = \mathbf{P}^T$  and  $\mathbf{A}_f$  is the fine mesh matrix.

The use of a coarse mesh to accelerate the convergence of a one-level Schwarz preconditioner is similar to multigrid methods that use a sequence of coarser meshes to accelerate the convergence of the solution on a fine mesh. Typically, more than two levels are employed in a multigrid approach. One of the disadvantages of a geometric multigrid approach is the need to generate a sequence of geometrical grids. Alternatively, one can use algebraic multilevel preconditioners, which do not require a sequence of coarser grids. First, a graph is built from the linear system. This graph contains an edge between two vertices  $i$  and  $j$ , if  $A_{i,j} \neq 0$ . For systems of PDEs, it is often natural to define this graph in a block fashion. That is, one vertex is associated with each block of unknowns (e.g. velocities and pressure at a particular grid point) and an edge between two vertices  $i$  and  $j$  is added if there are any nonzeros in the block matrix defined by the  $i^{th}$  block row and  $j^{th}$  block column. The basic characteristic of all algebraic multigrid methods is to coarsen the graph representation of the matrix. There are many ways to define coarse meshes and grid transfers. Perhaps the most well-known is the classical AMG approach of Ruge-Stüben [23] where a subset of fine mesh vertices are used to define the next coarser mesh. In this paper, an alternative scheme is employed where fine mesh vertices are grouped in aggregates. Each aggregate effectively represents a coarse grid vertex. Once the coarse mesh is determined, a grid transfer must be defined. The simplest possible grid transfer is to use piecewise constant interpolation. In this case, the grid transfer,  $P$ , contains only zeros and ones. In the scalar PDE case,  $P(i, j)$  is equal to one only if the  $i^{th}$  fine grid point has been assigned to the  $j^{th}$  aggregate. Within a PDE system, the grid transfer is a block system where a small identity matrix is inserted within the  $(i, j)^{th}$  block if the  $i^{th}$  fine grid point has been assigned to the  $j^{th}$  aggregate.

There are many ways to define aggregates. In a standard algebraic multigrid method the most common way to create an aggregate is via some kind of greedy graph algorithm where an initial node is chosen along with all of its nearest neighbors. The net affect of this type of procedure is to produce

aggregates which are “sphere-like” with an approximate diameter of three nodes. Unfortunately, this type of procedure leads to many aggregates and must be repeated several times in order to obtain a coarse matrix that is small enough to be efficiently solved by a direct solver. Since our goal is to mirror a domain decomposition method where only a couple of coarse meshes are used, we consider a more aggressive coarsening scheme that produces larger aggregates and is better suited for parallel computations. Specifically, the graph partitioning codes METIS and ParMETIS [24] are used to define the aggregates. The algorithms within these packages split the fine matrix so that each partition has no more nodes than a user supplied parameter. All the nodes in each partition are then combined to form a single aggregate. METIS is a serial code and so can only be used to partition the graph corresponding to the local matrix entries within a processor. This implies that no aggregate can span more than one processor. ParMETIS is a parallel code and so it can be applied to the entire distributed global graph to produce aggregates that span more than one processor.

The procedure we have outlined is sometimes referred to as *non-smoothed aggregation*. All the results presented in this paper will refer to nonsmoothed aggregation. In fact, even if for elliptic problems the so-called *smoothed aggregation* performs significantly better than nonsmoothed [5, 25], the correct smoothing procedure for nonsymmetric operators is still an open problem. Often, non-smoothed aggregation avoids stability problems that arise when smoothed aggregation is used. Further analysis on this subject is reported in [26].

For all preconditioners presented in this paper, the ratio between the size of the matrices for two consecutive levels is rather high, e.g. of order 500 for two-level methods and of order 100 for three-level methods. Because of this, often more effective (and heavyweight) smoothers are required, as lightweight smoothers, typically used in multigrid applications, may be ineffective.

The aggregation scheme and construction of restriction and prolongation operators is implemented within the ML library [22]. Support for the coarse matrix direct solvers KLU [27] and distributed version of SuperLU [28] are provided through the Amesos interface [27]. ML, Amesos, and Aztec are available through the Trilinos framework [29].

## 5. Results and Discussion

Algorithmic scaling studies comparing the one-level additive Schwarz preconditioner and two-level Schwarz preconditioner with geometric coarse operator were presented in a previous work [6]. These previous results demonstrated that the two-level Schwarz preconditioner with geometric coarse operator provided a one to two order of magnitude reduction in solution time in two dimensions and a factor of 5–8 reduction in solution time in three dimensions for a standard CFD benchmark problem of thermal convection in a square or cube geometry. The present work considers algorithmic scaling studies for the two- and three-level Schwarz preconditioner using an algebraic coarse grid applied to the following problems:

- 3D thermal convection problem
- Navier-Stokes flow in a 3D building geometry
- convection-diffusion equation.

On a subset of these problems we also compare these algebraic multilevel preconditioners with the one-level DD and the two-level geometric preconditioner.

For all the studies, two different example problems will be used. For the first example problem, previously used in [6], a thermal convection (or buoyancy-driven) flow in a differentially heated 1x1x1

cube in the presence of gravity is modeled. The momentum transport, energy transport and total mass conservation equations are solved. No-slip boundary conditions are applied on all walls. The temperature on the heated wall and other parameters are chosen so that the Rayleigh number is 1000. Figure 1a shows isosurfaces for x-component of velocity for a typical steady-state solution. This simple geometry facilitates the construction of regular-shaped subdomains as well as the coarse grid for the two-level preconditioner with geometric coarse operator. The second example involves the calculation of fluid flow, without thermal effects, in a simple prototype model of a building. This test case has been selected because the geometry and boundary conditions are representative of actual large-scale indoor structures. The dimensions of the 3D structure are 105 m x 24 m x 11 m. Inlets are located on the 24m sidewalls of both floors, the 105m length faces, the ceiling of the lower level, and the floor of the upper level. Outlets are located on the ceiling of the upper floor, with the main outlet collectors located above the atria between the two floors. Figure 1b shows a typical laminar steady-state solution for each problem. The centerline cutting plane shows the x-component of velocity. Unless otherwise stated, both test cases employ hexahedral meshes with bilinear finite elements.

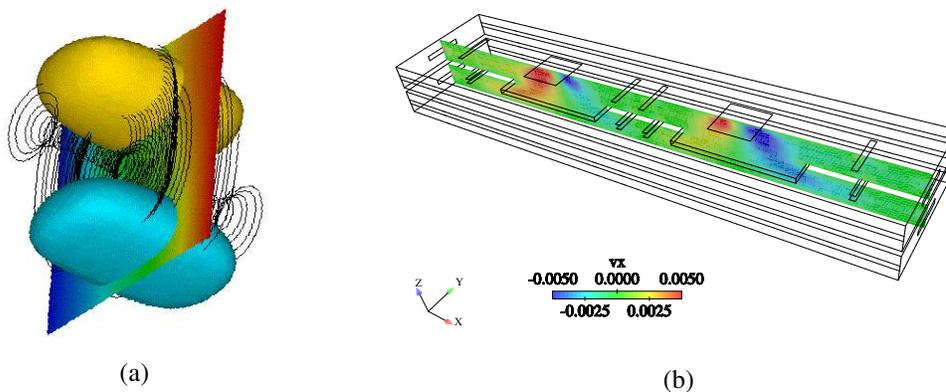


Figure 1. (a) Constant x-component of velocity isosurfaces with streamlines and temperature contours on slice plane for thermal convection problem with  $Ra=1000$ ; (b) Steady-state x-component of velocity on centerline cutting plane for model 3D building

### 5.1. Comparison between the two-level Schwarz preconditioner with geometric coarse operator and algebraic coarse operator

Table I presents the results for a scalability study that compares the two-level geometric coarse operator preconditioner with a two-level algebraic coarse operator preconditioner with nonsmoothed aggregation for the 3D thermal convection problem. The same fine mesh was used for both cases. The number of fine mesh nodes per aggregate was chosen in an attempt to produce coarse matrices of comparable sizes. The mesh for each successive row in the table is produced by a uniform refinement of the mesh in the previous row by cutting each cube into eight cubes. This roughly increases the number of unknowns by a factor of eight. The number of processors is also increased by a factor of eight so that the number of unknowns per processor stays roughly constant. For an algorithm with perfect scalability, both the number of iterations per Newton step and the CPU time should stay roughly constant. For most of the subsequent scaling studies, the sequence of meshes is produced in this fashion. “GS2/SuperLU” in columns 5-8 indicates that the fine mesh smoother employs two

proc	fine grid unks	coarse unknowns		2-level: GS2/SuperLU				2-level: GS2/GMRES-ILU			
				geometric		algebraic		geometric		algebraic	
		geom	alge	avg iter	time (s)	avg iter	time (s)	avg iter	time (s)	avg iter	time (s)
4	24.6K	135	120	33 [5]	78	30 [4]	71	33 [5]	94	30 [4]	69
32	180K	625	480	44 [4]	94	50 [4]	109	44 [4]	114	51 [4]	147
256	1.37M	3645	2560	47 [5]	219	58 [4]	152	47 [5]	238	58 [4]	196
2048	10.7M	24.6K	20.5K	ran out of memory				47 [4]	598	59 [4]	681

Table I. Comparison of geometric and algebraic two-level preconditioners for the 3D thermal convection problem.

sweeps of local Gauss-Seidel and the coarse mesh smoother uses the distributed SuperLU package [30]. Local Gauss-Seidel is a domain decomposition hybrid where each processor performs standard point Gauss-Seidel on its subdomain (as opposed to applying Gauss-Seidel to the entire global domain). Distributed SuperLU is a parallel direct solver package. “GS2/GMRES-ILU” in columns 9-12 denotes that the fine mesh smoother is two sweeps of local Gauss-Seidel and the coarse mesh smoother employs GMRES with an ILU preconditioner. The coarse grid GMRES is iterated to convergence upon each invocation of the preconditioner. This variant is needed to tackle the relatively large coarse mesh in the 2048 processor case. This coarse mesh is essentially the smallest size that is possible with our geometric coarse mesh software. Unfortunately, the parallel machine does not have sufficient memory for the direct solver to function. The column “avg iter” denotes the average number of iterations per Newton step; the second number in brackets following the average number of iterations is the number of Newton steps. The “time” column reports the CPU time for the steady-state solve. From these results, the two-level preconditioner with an algebraic coarse operator seems competitive with the two-level preconditioner with a geometric coarse operator. As a point of comparison, the one-level solver using a DD ILU preconditioner took 650 iterations per Newton step and 2915 seconds for the 10.7 million unknown 2048-processor case [6]. From Table I it is clear that the preconditioner with an algebraic coarse operator also maintains the optimal iteration count per Newton step, although the average iterations per Newton step tended to be slightly higher than with the geometric coarse operator. A possible explanation is that while the fine mesh has the same number of unknowns for the both cases, the two-level preconditioner with algebraic coarse operator has a coarse level with fewer unknowns than the preconditioner with geometric coarse operator. The number of iterations is the same when using either SuperLU or GMRES-ILU as the coarse iterative solve is iterated to convergence. However, the run time is noticeably longer using the coarse iterative solver. This is partially due to the direct solver’s reuse of the matrix factorization (that need only be computed once for each Newton step). It may be possible to improve the reuse capability within the iterative coarse solver (e.g. save and reuse Krylov vectors from the previous coarse solve) to make the times closer to that of the direct solver. These results were obtained on the ASCI-Red Tflop computer at Sandia National Laboratories, each node containing 256 MB RAM and 333 MHz Pentium II Xeon processors.

### 5.2. Algorithmic scaling studies for three-level Schwarz preconditioner with algebraic coarse operators for the 3D thermal convection problem

Scalability studies were performed for the 3D thermal convection cube problem for the three-level preconditioner for both the laminar steady-state and transient cases. Unless otherwise stated, for the

nodes per aggregate	proc	unknowns			avg its/ Newt step	time (sec)
		fine	medium	coarse		
50,200	2	180K	3590	15	49 [4]	488
	16	1.37M	27440	135	93 [4]	835
	128	10.7M	214K	1070	111 [4]	1191
	1024	84.9M	1.69M	8470	109 [4]	2050
100,100	2	180K	1790	15	52 [5]	615
	16	1.37M	13680	135	93 [4]	812
	128	10.7M	107K	1065	104 [4]	1101
	1024	84.9M	845K	8445	109 [4]	2117
200,50	2	180K	890	15	64 [4]	574
	16	1.37M	6800	135	95 [4]	844
	128	10.7M	53120	1060	111 [4]	1149
	1024	84.9M	420K	8395	119 [4]	2428

Table II. Scalability study of three-level preconditioner (GS1/ILU/SuperLU) for the steady 3D thermal convection problem (aggregation: METIS; ParMETIS); Cplant machine.

three-level preconditioner, the fine and medium mesh smoothers are one sweep of Gauss-Seidel and ILU respectively, and the coarse solver is the distributed version of SuperLU. These calculations, as well as the remaining calculations in the paper (with the exception of the calculations in Table V), were performed on the Sandia Cplant machine which is composed of 1024 nodes, each with 500 MHz Dec Alpha processor and 1 GB of RAM, connected together by Myrinet. The performance of each processor is very roughly comparable to a 1-GHz Intel Pentium III processor.

The first group of scalability studies concern the steady-state solution for the thermal convection problem with Rayleigh number of 1000. Table II shows an algorithmic scaling study for different numbers of nodes per aggregate. In the first column, the first and second number are the nodes per aggregate when constructing the medium level and coarse level respectively. "Avg its/Newt step" denotes the average number of iterations per Newton step, and the number of Newton steps is denoted by the number in brackets. The final column is the time for the steady-state solve. Within each group with the same number of nodes per aggregate, the smallest calculation is with a starting mesh that is run on two processors. As in the previous scaling study, the mesh is then uniformly refined three times, each level of uniform refinement increasing the number of hexahedra by a factor of eight, which increases the number of unknowns by a factor of roughly eight. The number of processors is also increased by a factor of eight so that the number of unknowns per processor stays roughly constant. After each level of uniform refinement, the fine mesh is load-balanced using Recursive Coordinate Bisection (RCB) through the Zoltan data management services for parallel applications package [31]. Aggregates are generated by resorting to local graph partitioning algorithms for the fine level (METIS) and a global graph partitioning algorithm (ParMETIS). Note that for the "50,200" nodes per aggregate case, the optimal convergence property is obtained (number of iterations per Newton step asymptotes to a fixed value), while the number of iterations per Newton step for the "100,100" and "200,50" nodes per aggregate case are clearly asymptoting to a fixed value. However, the CPU time shows a modest increase.

time step	approx max CFL	proc	unknowns			avg its/ Newt step	time (sec)
			fine	medium	coarse		
0.01	1	2	180K	1790	15	30	223
		16	1.37M	13680	135	48	330
		128	10.7M	107K	1065	59	527
0.1	10	2	180K	1790	15	44	296
		16	1.37M	13680	135	76	490
		128	10.7M	107K	1065	87	730
1.0	100	2	180K	1790	15	51	324
		16	1.37M	13680	135	90	568
		128	10.7M	107K	1065	103	852

Table III. Scalability study of three-level preconditioner (GS1/ILU/SuperLU) for the transient 3D thermal convection problem. METIS (ParMETIS) aggregation used on the fine (middle) mesh with 100 nodes per aggregate; Cplant machine.

The next group of algorithmic scaling studies concern transient flow for the 3D thermal convection problem. Table III shows the algorithmic scaling study. In this table, “avg its/Newt step” denotes the average number of iterations per Newton step and “time” denotes the average time per time step with both quantities calculated by averaging the first ten time steps. 100 nodes per aggregate with METIS and ParMETIS schemes for the first and second levels of aggregation respectively was used. Note that for each size of time step, the average iterations per Newton step is asymptoting to a fixed value. Although one further level of uniform refinement (which would be run on 1024 processors) would probably be needed to demonstrate the optimal convergence property, the trend is still clear.

### 5.3. Algorithmic scaling studies for three-level Schwarz preconditioner with algebraic coarse operators for Navier-Stokes flow in a 3D building geometry

As can be seen from the previous results, when the problem is scaled in size and proper number of processors to roughly maintain the same number of unknowns per processor, the three-level preconditioner obtains the optimal convergence property for a simple cube geometry. The next group of studies will investigate whether this optimal iteration convergence property can be maintained for a more realistic computational domain. Scalability studies were performed for the model 3D building for the three-level preconditioner for both steady-state and transient cases. Many of the same scalability studies performed with the 3D thermal convection problem are repeated for the 3D building, including the effect of varying the number of nodes per aggregate. As before, unless otherwise stated, the smoothers and solvers chosen for the three-level preconditioner are one sweep of Gauss-Seidel on the fine level, ILU on the medium level, and the distributed version of SuperLU for the coarse level.

For the steady-state calculations, the Reynolds number is restricted to two orders of magnitude lower than normal operating conditions based on an inlet duct reference length and velocity to provide a laminar flow. The low Reynolds number allows quick direct-to-steady-state solution and simplifies the presentation of the results. The transient turbulent large eddy simulation (LES) calculations are performed at the standard operating conditions of the ventilation systems. The 3D hexahedral meshes and tetrahedral meshes were generated by the Sandia Cubit mesh generation software [32].

Table IV shows an algorithmic scaling study for different numbers of nodes per aggregate for the steady-state calculations on hexahedral meshes. The larger meshes are generated by uniform refinement of previous meshes, with the number of processors being increased to maintain a roughly constant number of unknowns per processor. In contrast to the 3D thermal convection problem, after each level of uniform refinement of the building geometry, the fine mesh is load-balanced using the ParMetis graph partitioner through Zoltan. Note that in contrast to the thermal convection problem, the optimal iteration count is not obtained for the 3D building problem. This is most likely due to the fact that the combination of the aggregation scheme and the smoother is not handling hexahedral elements with medium or high aspect ratios properly. This is a well-known problem with multigrid methods where the performance of the smoother deteriorates in the “long” direction of elements with high aspect ratios. Remedies have been extensively studied in the context of structured meshes. These are primarily based on block relaxation techniques to improve smoothing in the long direction or semi-coarsening to coarsen less in the direction where smoothing is poor. This subject has been studied (though to a lesser extent) for algebraic methods and we are exploring possible enhancements to both our aggregation and smoothing methods to address this. For this particular problem, the hexahedral elements vary in aspect ratio with the worst aspect ratio elements having a longest dimension that is a factor of 5.3 larger than the shortest dimension. The hexahedral meshes used are actually structured meshes. Previous experience has shown that when the elements are close to squares and cubes for two and three dimensions respectively, then the optimal convergence property is obtained. Reference [33] notes the effect of the aspect ratio of the elements on algebraic methods and suggests a promising alternative approach.

Further insight into the problem can be gained by considering a 2D thermal convection problem with differentially heated walls (2D version of Figure 1a) in a rectangular domain that is sixteen times longer than it is high. The two-level preconditioner with geometric coarse mesh is used, with ILU smoother on the fine mesh and KLU [27] solver on the coarse mesh. The top half of Table V considers the discretization of the domain with square-shaped finite elements, so there are 16 times as many elements in the lengthwise direction. For each successive row, the mesh is refined by splitting a square element into four squares. Note that the optimal convergence property is obtained as the problem is scaled from 4 to 64 processors. The bottom half of the table considers the discretization of the domain with rectangular-shaped finite elements that are four times longer than they are high. The number of unknowns per processor is roughly the same as the corresponding row in the top half of the table. Note that the optimal iteration count is not obtained. This shows that even the preconditioner with geometric coarse mesh (where aggregation is not used) does not properly handle the medium aspect ratio finite elements. This is most likely an indication that the smoother does not sufficiently smooth error in the stretched direction. It also illustrates the difficulties in applying black-box solvers to a wide range of problems.

To further demonstrate that the issue of failure to obtain the optimal convergence property is with the aspect ratio of the finite elements and not the geometry, a scalability study was performed on a tetrahedral mesh of the 3D building where the tetrahedra are of relatively low aspect ratio. This tetrahedral mesh was generated from a triangular surface mesh by using Cubit’s volume mesh generator rather than taking the previous hexahedral mesh and cutting a hexahedron into six tetrahedra. In contrast to the hexahedral mesh case, the finer tetrahedral meshes were not created from uniform refinement of coarser tetrahedral meshes. They were created by initially reducing the interval size between nodes on the surface triangulation by half, with further reduction of interval size in order to give an increase in the number of nodes by a factor of 7–8. Table VI shows this algorithmic scaling study. It also shows a comparison between the one-level ILU preconditioner and the three-level

nodes per aggregate	proc	unknowns			avg its/ Newt step	time (sec)
		fine	medium	coarse		
50,200	2	227K	4544	20	38 [5]	458
	16	1.70M	33988	168	52 [5]	638
	128	13.1M	263K	1312	67 [5]	1142
	1024	103M	2.06M	10316	121 [5]	2656
100,100	2	227K	2272	20	40 [5]	466
	16	1.70M	16976	168	53 [5]	629
	128	13.1M	131K	1308	76 [5]	1160
	1024	103M	1.03M	10304	136 [5]	2924
200,50	2	227K	1136	20	43 [5]	482
	16	1.70M	8480	168	60 [5]	689
	128	13.1M	65460	1308	89 [5]	1344
	1024	103M	514K	10284	160 [5]	3479

Table IV. Scalability study of three-level preconditioner (GS1/ILU/SuperLU) for the steady 3D building problem with hexahedral mesh (aggregation: METIS, ParMETIS); Cplant machine.

element aspect ratio	proc	fine mesh	coarse mesh	avg its/ Newt step	time (sec)
1:1	4	257 × 17	33 × 3	29	15
	16	513 × 33	65 × 5	31	22
	64	1025 × 65	129 × 9	32	48
4:1	4	129 × 33	17 × 5	55	23
	16	257 × 65	33 × 9	135	65
	64	513 × 129	65 × 17	convergence failed	

Table V. Scalability study of two-level preconditioner (ILU/KLU) with geometric coarse mesh for the steady 2D thermal convection problem; ASCI Red machine.

preconditioner. Note that the three-level preconditioner provided a factor of four reduction in time over the one-level preconditioner for the 128 processor case. The number of iterations per Newton step for the one-level preconditioner scales as roughly the number of unknowns to the power of  $\frac{1}{3}$  while for the three-level preconditioner seems to be asymptoting to a constant value. This demonstrates that the optimal convergence property is obtained when good aspect ratio finite elements are used. We are currently working on a fix to this problem so that the optimal convergence property will be obtained for both medium and high aspect ratio finite elements.

The next group of algorithmic scaling studies concern transient flow in the 3D building. The large eddy simulation (LES) model used was the LES-k model [34]. Table VII shows the algorithmic scaling study. Two different time steps were used:  $\Delta t = 0.1$  seconds and  $\Delta t = 1.0$  seconds. In this table, “avg its/Newt step” denotes the average number of iterations per Newton step and “time” denotes the average time per time step with both quantities calculated by averaging the first ten time steps. 100

proc	fine	medium	coarse	1-level		3-level	
				avg its/ Newt step	time (sec)	avg its/ Newt step	time (sec)
2	227K	2272	20	83 [5]	910	35 [5]	454
16	1.68M	16832	168	148 [6]	1571	56 [5]	605
128	13.1M	131K	1308	302 [6]	4099	51 [6]	1000

Table VI. Scalability study of 1-level (ILU) and three-level preconditioner (GS1/ILU/KLU) for the steady 3D building problem with tetrahedral mesh (aggregation: METIS, ParMETIS); Cplant machine.

time step	approx max CFL	proc	unknowns			avg its/ Newt step	time (sec)
			fine	medium	coarse		
0.1	0.03	2	224K	2230	20	71	530
		16	1.67M	16670	165	72	501
		128	12.9M	129K	1285	75	619
1.0	0.3	2	224K	2230	20	46	322
		16	1.67M	16670	165	52	371
		128	12.9M	129K	1285	58	574

Table VII. Scalability study of three-level preconditioner (GS1/ILU/SuperLU) for the transient 3D building problem (LES-k) with hexahedral mesh. METIS (ParMETIS) aggregation used on the fine (middle) mesh with 100 nodes per aggregate; Cplant machine.

nodes per aggregate with METIS and ParMETIS schemes for the first and second levels of aggregation respectively was used. The scaling in Table VII is much better than that exhibited in the steady-state study even though similar quality hexahedral meshes were used. Though further analysis is needed, we conjecture that the lumped mass matrix term may offset the suboptimal behavior observed for medium aspect ratio hexahedral mesh steady-state problems.

#### 5.4. Comparisons between one-level, two-level, and three-level preconditioners for fluid flow in a 3D building geometry

Table VIII compares various one-level, two-level, and three-level preconditioners for steady laminar flow in the 3D model building with Reynolds number two orders of magnitude lower than actual conditions. This geometry has 2.6 million grid points (10.3 million unknowns) and 2.5 million hexahedral elements. These cases were run on 128 nodes of the Sandia Cplant machine. For this test case, the two-level preconditioner with geometric coarse operator performed marginally better than the one-level preconditioner. This is mainly due to the substantial cost of the direct solve on the coarse mesh. The better CPU time of the algebraic two-level preconditioner compared with the geometric preconditioner is likely due to the smaller coarse matrix. The algebraic three-level preconditioner performed well due to the fact that an inexpensive smoother such as Gauss-Seidel could be used on the fine mesh.

Table IX presents a comparison between various one-level, two-level, and three-level preconditioners

method	smoothers/ solver	nodes per aggregate	unknowns		avg its/ Newt step	time (sec)
			medium	coarse		
1-level DD	ILU				188 [5]	3074
2-L geom	ILU/SuperLU			25520	38 [5]	2694
2-L geom	GS2/SuperLU			25520	did not converge	
2-L alge	ILU/SuperLU	512		19948	37 [5]	2104
2-L alge	GS2/SuperLU	512		19948	237 [5]	5918
3-L alge	GS2/ILU/SuperLU	64	201K	1256	85 [5]	1419
3-L alge	GS2/GS2/SuperLU	64	201K	1256	103 [5]	1604

Table VIII. Comparison of different preconditioners for the 3D model building; steady-state; fine mesh with 10.3 million unknowns; 128 processors on Cplant machine.

for a transient LES simulation in the 3D model building with realistic Reynolds number based on inlet dimensions and inlet velocity. This geometry has 3.3 million nodes (13.1 million unknowns) and 3.2 million hexahedral elements. The size of the time step is  $\Delta t = 0.1$  seconds and the approximate maximum CFL on the fine grid is 0.01. These cases were run on 1000 nodes of the Sandia Cplant machine. Because of limited access to 1000 nodes of the machine, only the first four Newton steps of the first time step were run for comparison. Note that the two-level preconditioner with geometric coarse operator did worse than the one-level preconditioner. This is likely due to the expensive coarse grid solve and the relatively good convergence obtained by the one-level preconditioner when a well-conditioned transient system is solved. The algebraic three-level preconditioner did substantially better than the one-level or two-level preconditioners because the use of three levels allowed an inexpensive smoother such as Gauss-Seidel to be used on the large fine mesh. This becomes a significant advantage as the size of the fine mesh increases, especially when a two-level preconditioner will not converge with an inexpensive smoother on the fine mesh and an expensive smoother such as ILU is required. This was very apparent when a steady-state laminar calculation (with Reynolds number two orders of magnitude too low) was performed for the 3D model building with 25.8 million nodes (103 million unknowns) on 1000 nodes of the Cplant machine. Both the one-level and two-level methods require an expensive fine mesh smoother (ILU), and after 3 hours, the solver was still computing the first ILU factorization. The algebraic three-level preconditioner with Gauss-Seidel, ILU, and SuperLU on the fine, medium, and coarse mesh respectively reached steady-state after 75 minutes. Note that in the three-level preconditioner scaling studies, with a choice of 50 and 200 nodes per aggregate for first and second levels of aggregation, a steady-state solution could be obtained in 45 minutes.

##### 5.5. Algorithmic scaling studies for three-level Schwarz preconditioner with algebraic coarse operators for the solution of the convection-diffusion equation

Scalability studies were performed for a single convection-diffusion equation, with the three-level Schwarz preconditioner. For the three-level preconditioner, the fine and medium mesh smoothers are one sweep of Gauss-Seidel and ILU respectively, and the coarse solver is the KLU direct solver. METIS and ParMETIS were used for aggregation for the fine and middle mesh respectively. A flow field that was obtained from a prior steady-state laminar Navier-Stokes calculation in the model 3D building provides the velocity field for the transient convection-diffusion equation calculation. Table X shows a

method	smoothers/ solver	nodes per aggregate	unknowns		avg its/ Newt step	time (sec)
			medium	coarse		
1-level DD	ILU				113	150
2-L geom	ILU/GMRES-ILU			32336	24	255
3-L alge	GS/ILU/SuperLU	100	129K	1292	31	38
3-L alge	GS/ILU/SuperLU	512	20376	44	56	53

Table IX. Comparison of different preconditioners for 3D model building; transient LES; fine mesh with 13.1 million unknowns; 1000 processors on Cplant machine

time step	proc	unknowns			1-level		3-level			
		fine	medium	coarse	avg its/ Newt step	time (sec)	GS/ILU/KLU		ILU/ILU/KLU	
							avg its/ Newt step	time (sec)	avg its/ Newt step	time (sec)
0.1	2	45K	446	4	2	9	3	9	—	—
	16	334K	3334	33	2	15	3	15	2	15
	128	2.58M	26K	257	2	21	4	20	—	—
1.0	2	45K	446	4	2	9	4	10	—	—
	16	334K	3334	33	2	15	5	15	2	15
	128	2.58M	26K	257	3	21	6	21	—	—

Table X. Scalability study of three-level preconditioner (GS1/ILU/KLU and ILU/ILU/KLU) for solution of a convection-diffusion equation in the 3D building. METIS (ParMETIS) aggregation used on the fine (middle) mesh with 100 nodes per aggregate; Cplant machine.

comparison between the one-level Schwarz preconditioner and the three-level Schwarz preconditioner for different sized time steps. The transient calculation was run for ten time steps. The reported “average iterations per Newton step” and “time” is the averaged iterations per Newton step and time over the ten time steps. Note that the one-level preconditioner required fewer iterations than the 3-level preconditioner (when Gauss-Seidel is the smoother on the fine mesh). This is due to the fact that this problem is very well-conditioned and therefore the one-level method performs extremely well. As presented in the table for the 16 processor case, the use of ILU rather than one sweep of Gauss-Seidel as the fine mesh smoother for the three-level preconditioner would yield the same iteration count as for the one-level preconditioner. This demonstrates that for simple, well-conditioned problems, the one-level ILU preconditioner performs well and there is no advantage to using the multilevel preconditioner.

## 6. Conclusions

This study compares, for low to medium Reynolds number Navier-Stokes flows, several preconditioners based on domain decomposition and multilevel schemes, including one-level additive Schwarz, a two-level Schwarz with geometric coarse operator, and two- and three-level Schwarz

based on aggregation. Except for trivial problems, as expected, the one-level preconditioner is not scalable. The two-level preconditioners, one based on a geometric coarse operator and the other based on an algebraic coarse operator, are scalable for medium-size problems, and have comparable performance. For large-size problems the CPU-time required to solve the coarse level problem becomes predominant, and the preconditioner fails to be scalable. By resorting to three-level preconditioners based on aggregation, the optimal iteration count for a 3D benchmark thermal convection problem hexahedral meshes was obtained as well as for the 3D model building problem with tetrahedral meshes. Results demonstrate that often the use of more than two levels for the algebraic preconditioner resulted in faster CPU times. Work is currently in progress to allow the optimal convergence property to be obtained for medium and high aspect ratio finite elements. Further study is needed to compare how the preconditioners perform for highly convective flows and for reactive flows.

#### ACKNOWLEDGEMENTS

The authors would like to thank Gary Hennigan, Jonathan Hu, Karen Devine, and Mike Heroux for their contributions to this work and to Andrew Salinger and Roger Pawlowski for continued development of the MPSalsa code. The authors would like to express their gratitude to the Cplant management team for providing the dedicated run time as well as to the Cplant operations team led by Sophia Corwell for providing superb support during the periods of dedicated run time.

#### REFERENCES

1. W.D. Gropp, D.K. Kaushik, D.E. Keyes, and B.F. Smith. High-performance parallel implicit CFD. *Parallel Computing*, 27:337–362, 2001.
2. B. Smith, P. Bjorstad, and W. Gropp. *Domain Decomposition: Parallel Multilevel methods for elliptic partial differential equations*. Cambridge University Press, 1996.
3. Ulrich Trottenberg, Cornelis Oosterlee, and Anton Schüller. *Multigrid*. Academic Press, London, 2001.
4. M. Brezina and P. Vaněk. A black-box iterative solver based on a two-level Schwarz method. *Computing*, 63:233–263, 1999.
5. P. Vanek, M. Brezina, and J. Mandel. Convergence of algebraic multigrid based on smoothed aggregation. *Numerische Mathematik*, 88:559–579, 2001.
6. J.N. Shadid, R.S. Tuminaro, K.D. Devine, G.L. Hennigan, and P.T. Lin. Performance of fully-coupled domain decomposition preconditioners for finite element transport/reaction simulations. *Accepted for publication in Journ Comp Phys*.
7. L. Jenkins, T. Kelley, C.T. Miller, and C.E. Kees. An aggregation-based domain decomposition preconditioner for groundwater flow. Technical Report TR00–13, Department of Mathematics, North Carolina State University, 2000.
8. C. Lasser and A. Toselli. An overlapping domain decomposition preconditioner for a class of discontinuous Galerkin approximations of advection-diffusion problems. *Mathematics of Computation*, (72):1215–1238, 2003.
9. M. Sala and L. Formaggia. Algebraic coarse grid operators for domain decomposition based preconditioners. In P. Wilders, A. Ecer, J. Periaux, N. Satofuka, and P. Fox, editors, *Parallel Computational Fluid Dynamics – Practice and Theory*, pages 119–126. Elsevier Science, The Netherlands, 2002.
10. M. Sala. *Domain Decomposition Preconditioners: Theoretical Properties, Application to the Compressible Euler Equations, Parallel Aspects*. PhD thesis, EPFL, Lausanne, Switzerland, 2003.
11. T.J.R. Hughes, L.P. Franca, and M. Balestra. A new finite element formulation for computational fluid dynamics: V. circumventing the Babuska-Brezzi condition: a stable Petrov-Galerkin formulation of the Stokes problem accommodating equal-order interpolations. *Comp. Meth. AMech. and Eng.*, 59:85–99, 1986.
12. T.E. Tezduyar. Stabilized finite element formulations for incompressible flow calculations. *Advances in App. Mech.*, 28:1–44, 1992.
13. J.N. Shadid, S.A. Hutchinson, G.L. Hennigan, H.K. Moffet, K.D. Devine, and A.G. Salinger. Efficient parallel computation of unstructured finite element reacting flow solutions. *Parallel Computing*, 23:1307–1325, 1997.
14. P.N. Brown and Y. Saad. Hybrid Krylov methods for nonlinear systems of equations. *SIAM J. Sci. Stat. Comp.*, 11:450–481, 1990.

15. J.N. Shadid. A fully-coupled Newton-Krylov solution method for parallel unstructured finite element fluid flow, heat and mass transfer simulations. *Int. J. CFD*, 12:199–211, 1999.
16. S.C. Eisenstat and H.F. Walker. Globally convergent inexact Newton methods. *SIAM J. Optimization*, 4:393–422, 1994.
17. Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, 2003.
18. A. Quarteroni and A. Valli. *Domain Decomposition Methods for Partial Differential Equations*. Oxford University Press, Oxford, 1999.
19. Y. Saad. ILUT: a dual threshold incomplete ILU factorization. *Numerical Linear Algebra with Applications*, 1:387–402, 1994.
20. S.A. Hutchinson, L. Prevost, J.N. Shadid, C. Tong, and R.S. Tuminaro. Aztec user's guide version 2.0. Sandia National Laboratories Technical Report SAND 99-8801J, 1999.
21. R.S. Tuminaro, C.H. Tong, J.N. Shadid, K.D. Devine, and D.M. Day. On a multilevel preconditioning module for unstructured mesh Krylov solvers: two-level Schwarz. *Comm. Num. Method. Eng.*, 18:383–389, 2002.
22. M. Sala, J. Hu, and R. Tuminaro. ML 3.1 smoothed aggregation user's guide. Technical Report SAND2004-4819, Sandia National Laboratories, September 2004.
23. J.W. Ruge and K. Stüben. Algebraic multigrid. in *Multigrid Methods*, Vol.3, *Frontiers in Applied Mathematics* 73-130, 1987.
24. G. Karypis and V. Kumar. Multilevel k-way partitioning scheme for irregular graphs. Army HPC Research Center Technical Report 95-064, 1995.
25. M. Brezina. *Robust iterative method on unstructured meshes*. PhD thesis, University of Colorado at Denver, 1997.
26. M. Brezina, M. Sala, R. Tuminaro, and J. Shadid. Aggregation for nonsymmetric systems. *In preparation*, 2004.
27. M. Sala. Amesos 2.0 reference guide. Technical Report SAND2004-4820, Sandia National Laboratories, September 2004.
28. J. W. Demmel, J. R. Gilbert, and X. S. Li. Superlu users' guide. Technical Report LBNL-44289, Lawrence Berkeley National Laboratory, September 1999.
29. Michael Heroux, Roscoe Bartlett, Vicki Howle Robert Hoekstra, Jonathan Hu, Tamara Kolda, Richard Lehoucq, Kevin Long, Roger Pawlowski, Eric Phipps, Andrew Salinger, Heidi Thornquist, Ray Tuminaro, James Willenbring, and Alan Williams. An Overview of Trilinos. Technical Report SAND2003-2927, Sandia National Laboratories, 2003.
30. X.S. Li. An overview of SuperLU: Algorithms, implementation, and user interface. Technical Report LBNL-53848, 2003.
31. 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.
32. Steven J. Owen ed. Cubit 9.0 users manual. Sandia Report SAND94-1100 Rev. 4/2002, Sandia National Laboratories, June 2004.
33. E. Chow. An unstructured multigrid method based on geometric smoothness. *Numer. Linear Algebra Appl.*, 10:401–421, 2003.
34. J. N. Shadid, A. G. Salinger, R. C. Schmidt, S. A. Hutchinson, G. L. Hennigan, K. D. Devine, and H. K. Moffat. MPSalsa: A finite element computer program for reacting flow problems part I: Theoretical development. Technical Report SAND98-2864, Sandia National Laboratories, Albuquerque NM, 87185, January. 1999.