Parallel Strategies For Algebraic Multi-Grid

XIANG ZHAO¹, KUNAL JAIN² and SIJUN ZHANG²

¹College of Arts and Sciences, Athens State University, 300 North Beaty St, Athens, AL 35611, USA
²ESI-CFD, Inc., 6767 Old Madison Pike, STE 600, Huntsville, AL 35806, USA

Abstract: The major portion of computing time in a computational fluid dynamic (CFD) flow solver is consumed by the solution of the linear equations, the use of an efficient matrix solver is critical to obtain high computing performance. Currently, Algebraic Multi-Grid (AMG) is recognized in CFD community as one of the best choices to achieve high efficiency for solving linear algebraic equations. As AMG is increasingly used for large problems, its parallelization is highly demanded. In this paper, a parallel algorithm for AMG is proposed with regard to unique characteristics of partial differential equations governing fluid flows. A building strategy was developed to accomplish grid coarsening and build the hierarchy of multi-level grids. Moreover an agglomeration approach was proposed, which comes from the physical and mathematical properties of flow fields. The parallelized AMG has demonstrated excellent performances on many cases.

Key-Words: Computational fluid dynamics, Multi-level grid, Parallel algorithms, Numerical algorithms, Parallel computing, Algebraic multi-grid (AMG)

1 Introduction

The multi-grid method has been demonstrated as an efficient means for obtaining solutions to fluid flows. In this approach, convergence acceleration is achieved by iterations on successively coarser meshes. The principle behind this algorithm is that the errors associated with the high frequencies are damped by a carefully chosen smoother, whereas the errors associated with the low frequencies are damped on the coarser grids where these frequencies manifest themselves as high frequencies. In the case of structured grid, coarser grids are easily derived from a given fine grid by omitting alternated grid lines in each coordinate direction, in which it is called geometric multigrid (GMG) [1~3]. In the case of unstructured grids, three different approaches can be adopted. Instead of GMG in structured grid method, it is named as AMG [4~6]. AMG should not be regarded as a competitor of GMG but rather as an efficient and flexible alternative to popular one-level methods. At present, it is widely used in computational mechanics including fluid and solid.

AMG is similar to conjugate gradient (CG) algorithms in that both are subspace methods; both look for the best solution in a subspace of the original problem. The difference is that CG algorithms use a larger subspace every iteration, and so are guaranteed to get the solution in N steps, where N is the number of equations. AMG uses the same subspace every iteration, and so needs another ingredient to remove the error which doesn’t lie in the subspace. This is the role of the smoother. Numerically, several choices are available for the smoother AMG, they include symmetric Gauss-Seidel (SGS), incomplete lower upper decomposition (ILU), conjugate gradient squared (CGS), Bi-conjugate gradient (Bi-CG), Bi-conjugate gradient square stabilized (Bi-CGSTAB) and generalized minimal residual (GMRES) [7]. After widely testing, it was found that GMRES is most robust among these smoothers. Theoretically, GMRES may be the only iterative matrix solver currently available which can not break down, regardless of the positiveness of matrix. However, the computer memory may increase dramatically based on the property of the matrix. GMRES is widely recognized as the most robust in matrix solvers today.

While sequential AMG has been used for increasingly large problems, its application on much larger practical cases needs it to be parallelized. Generally, there are two schemes for matrix solver parallelization, explicit and implicit. Explicit schemes are relatively easy to parallelize. As all operations are performed on data from preceding time steps it is only necessary to exchange the data at the interface regions between neighboring subdomains after each step is completed. The sequence of operations and results are identical on one and many processors. The most difficult part of explicit scheme is usually the solution of the elliptic Poisson-like equation for the pressure or pressure-correction equation. Implicit methods are more difficult to
parallelize. While calculation of the coefficient matrix and source vector uses only ‘old’ data and can be efficiently performed in parallel, the solution of the linear equation systems is not easy to parallelize.

Normally, the solution information is swapped at each grid level among adjacent domains; I found that it is not sufficient for high performance and robustness with parallel AMG. Mathematically, the solution of flow field consists of a series of waves, the domain-based information can locally determine the flow solution, and however the flow solution essentially depends on global information, i.e. whole flow field. Therefore, with the explicit block coupling in parallel computing, the solution may diverge in some cases. Thus, in the present work the agglomeration for coarsest grid is carried out to handle complex problems. During the restriction process of AMG procedures, data communication for intermediate solutions among processors are performed to synchronize the solutions for the whole flow field. At the bottom, i.e. coarsest grid, the grids and solutions for all domains are combined into a single zone and GMRES is applied to the combined zone. Then solutions are redistributed to corresponding sub-domains. This algorithm proved very effective for some unstable or poor convergence cases as we encountered in some cases.

2 Coarsening Algorithms

The original motivation for GMG was based on an important observation: iterative solvers tend to reduce short wavelength errors preferentially over long wavelength errors, in which short and long are defined with respect to the grid spacing. The hierarchy of successively coarser grids is designed so that various error components can be reduced. Unlike coarsening procedure in GMG, the procedure in AMG is not trivial and quite complicated. The primary difficulty for the multi-grid on unstructured grids is the creation and use of coarse grid hierarchy. On a structured grid, the coarse grids can be formed simply by removing every other grid line from the fine grids and the prolongation and restriction operators are simple to formulate. Among the methods for coarsening with AMG, the agglomeration is most popular and widely used in CFD. The major techniques are described as follows.

The discretization of the governing equations forms a system of linear equations and each equation contains the flow variables in a cell and its neighbor cells in the grid. The linear equation at a cell $i$ can be written in the form

$$a_i \phi_i - \sum_{nb\ of\ i} a_{i nb} \phi_{nb} = b_i$$  \hspace{1cm} (1)

where $nb$ means the neighbor cell of $i$. Assume the solution on the next coarser mesh cell $I$ which contains the next finer mesh cell $i$ is $\Phi_{i/I}$. The correction on the next finer mesh from the next coarser mesh is

$$\phi_i = \phi_{I} + \Phi_{i/I}$$  \hspace{1cm} (2)

In order to enforce the residual sum in $I$ as zero,

$$\sum_{i=1}^{N_B} r_i = 0$$  \hspace{1cm} (3)

The equation on the next coarser mesh comes out as

$$\sum_{i=1}^{N_B} a_i \Phi_{i/I} - \sum_{nb\ of\ i} a_{i nb} \Phi_{nb} = \sum_{i=1}^{N_B} r_i$$  \hspace{1cm} (4)

Rewrite the equation on the next coarser mesh in the form of

$$A_i \Phi_{I} - \sum_{nb\ of\ i} A_{i nb} \Phi_{nb} = B_i$$  \hspace{1cm} (5)

Then the $A_i$, $A_{i nb}$ and $B_i$ come directly from Eqn. (3) as

$$A_i = \sum_{i=1}^{N_B} a_i - \sum_{nb\ of\ i} a_{i nb}$$  \hspace{1cm} (6)

$$A_{i nb} = \sum_{i=1}^{N_B} a_{i nb}$$  \hspace{1cm} (7)

$$B_i = \sum_{i=1}^{N_B} r_i$$  \hspace{1cm} (8)

The algebraic multi-grid (AMG) method contains restriction and prolongation processes. In the restriction process, the equation coefficients and source terms on coarse mesh are generated based on those on fine mesh from Eqns. (6–8). While in the prolongation process, variables on fine mesh are modified by those on coarse mesh according to Eqn. (2). It can be seen that there is no actual coarse grid mesh in algebraic multi-grid method, but only the coarse grid equations followed completely from the fine grid equations. This solver strongly based on physical conservative concept rather than pure mathematical property. It requires the governing equation on fine grid to be in conservation form.

The coarsening process is merging the adjacent cells to form larger finite volume with the limitation on minimum and maximum number of cells to group together. The picked merging cells are those that have strong connection with the cells in the group. The neighbor cell $n$ is considered as strongly connected with cell $i$ if

$$a_i^n > c \cdot a_i^{\max}$$  \hspace{1cm} (9)

where $a_i^{\max}$ presents the largest neighboring
coefficient among $a_{nb}$, $c$ is a constant and the value of 1/3 works well. The minimum and maximum fine cell number in one coarse volume is set to 5 and 9 for 2-D, and 9 and 13 for 3-D, respectively.

A multi-grid cycle can be considered as a recursive procedure that is applied at each grid level as it moves through the grid hierarchy. Generally, there are four types of multi-grid cycles: V, W, F and flexible cycles. The paths of V cycle and W cycle are shown in Fig. 1, where the “R” and “P” stand for restriction and prolongation processes, respectively, and “S” is the solution. The W cycle is more efficient to get solutions because each grid level has the chance to pass its residual down to the coarse grid level twice and receive the corrections twice.

![V cycle and W cycle diagram](image)

Fig. 1 V cycle and W cycle diagram

The multi-grid F cycle is essentially a combination of the V and W cycles. An F cycle is formed by a W cycle followed by a V cycle. As expected, the F cycle requires more computation properties turn out to be better the W cycle. However, its convergence performances are better than the V cycle and roughly equivalent to the W cycle.

For the flexible cycle, the calculation and use of coarse grid correction is controlled in the multi-grid procedure by the logic switch. This logic switch ensures that coarser grid calculations are invoked when the rate of residual reduction on the current grid level is too slow. In addition, the multi-grid controls dictate when the iterative solution of the correction on the current coarse grid level is sufficiently converged and should thus be applied to the solution on the next finer grid. Note that the logic switch of the multi-grid procedure is such that grid levels may be visited repeatedly during a single global iteration. The main difference between the flexible cycle and the V and W cycles is that the satisfaction of the residual reduction tolerance and termination criterion determine when and how often each level is visited in the flexible cycle, whereas in the V and W cycles the traversal pattern is fixed.

3 Agglomeration Strategy

The primary algorithms for parallelization of AMG [8] has been developed to accelerate the solution process for large problems, in which the implicit method was used to parallelize the sequential AMG to ensure the stability and the convergence of the solutions.

In the parallel AMG method, all variables are partitioned into a number of subsets and assigned to the processors in the parallel system. For level $k$ in the AMG virtual grid hierarchy constructed by the steps described above, $\phi^k$ stands for the total number of variables. Assuming $k$ is not the coarsest level, we have

$$\phi^k = \bigcup_{p=1}^{P} \phi_p^k$$

(10)

where $P$ is the number of processors, $\phi_p^k$ is the subset of variables assigned to processor $p$. For $k = 1$, the partitioning of variables is approximately load-balanced among processors. Then each processor executes the steps of the AMG method in parallel on its own subset of variables. During the restriction process, each processor performs virtual grid coarsening independently of the other processors. Data communication for intermediate solution exchange among processors is performed to synchronize the solutions for the whole flow field. At the coarsest grid, the grids and solutions for all subdomains are combined into a single zone (also called final agglomeration) and GMRES algorithm is applied to the combined zone. Then solutions are redistributed to corresponding subdomains.

Similarly, during the prolongation process, each processor virtually refines the grid independently. Each processor $p$ needs to access any cells of its
neighboring processor \( q \) that are located inside the interpolation area. The interpolation area is the region that lies between \( p \) and \( q \) and contains the cells involved in the interpolations during the prolongation process. So the prolongation process also requires data communication between adjacent subdomains. Because of the independent coarsening and refinement, the partitioning of variables may not be load-balanced on all processors for level \( k \) \((k > 1)\).

A reasonable treatment of the coarsest AMG level is particularly important. In parallelizing GMG methods that are applied to relatively simple grids, we usually either stop coarsening whenever the number of variables per processor is too small or continue coarsening, with more and more processors idle. However, we use an intelligent agglomeration strategy to treat the coarsest AMG level for better efficiency.

Towards coarser levels, the parallel grid coarsening generally causes a slower reduction in the number of variables than sequential coarsening. Therefore, at some coarse level \( m_0 \), it becomes inefficient to continue parallel coarsening on all processors. Instead, it is more efficient to perform some agglomeration process for \( m > m_0 \). That is, the coarsening continues, but more and more subdomains are joined and treated by fewer and fewer processors. If only one processor is left, we solve the respective equations by a sequential GMRES solver. The selection of \( m_0 \) is crucial to the performance of the parallel AMG method. If agglomeration starts too early, the resulting coarse-level load imbalance may adversely influence the overall load balance; if it starts too late, the work on the coarsest levels may become unnecessarily high. Therefore, we develop a routine to monitor and compare the workload on the processors in the system. Based on the workload information, we determine when to start the agglomeration process; this is the key point of our intelligent agglomeration strategy.

There are two drawbacks to simply stopping coarsening at level \( m_0 \) without explicitly performing agglomeration because even though the number of variables per processor becomes relatively small, the total number of variables is still large. In complex applications, the convergence of many parallel iterative methods to solve the level \( m_0 \) correction equations is hard to predict. It is often the case that the cost on level \( m_0 \) becomes a substantial fraction of the total work load in the solution process, which reduces the overall performance.

### 4 Experimental Results

The proposed parallel strategy for AMG has been implemented in CFD-ACE+ and tested on a variety of benchmark and practical cases. All tests presented in this section were run on a PC cluster.

The first case is a subsonic flow past a three-element airfoil that was used by Lin and Dominick [9]. This is a good test case because it has undergone extensive laboratory testing in the Low Turbulence Pressure Tunnel located at NASA Langley Research Center and can be regarded as a benchmark [10]. This simple 2D geometry is selected as primary validation case to examine the coarsening strategy, the domain decomposition as well as the numerical operations such as restriction and prolongation that are used in our parallel AMG method. The flow is free-stream with Mach number of 0.2 and an angle of attack of 16.2 degrees. The computational grid contains 10,403 nodes and 20,294 triangles. The computed pressure contours for the entire airfoil are showed in Fig. 2(a), with the slat on the left side, chord in the middle, and flap on the right; Fig. 2 (b) shows part of the enlarged flow field around the slat and chord. The calculated surface pressure coefficient \( C_p \) is compared with experimental data in Fig. 3, which shows that our computational results agree well with the physical experimental results. Table 1 summarizes the speedup and efficiency for different number of processors. Our parallel AMG method achieves significant speedup over the sequential version and exhibits good parallel efficiency. Fig. 4 shows the ideal linear speedup and actual speedup. The speedup of our parallel AMG method increases as we add more processors.
method works well and achieves significant speedup over the sequential method in this case. Fig. 8 shows the speedup curves for the Boeing 747 case with or without using our agglomeration strategy. We can see our agglomeration increases the speedup of our parallel method by 5% on average. There are several factors that adversely affect the improvement of speedup using the agglomeration strategy. First, the load imbalance introduced by the coarsening process may become worse as subdomains merge, and thus degrade the parallel performance of our AMG method. Second, the overhead for subdomain merging also adversely affects the overall performance. Further research work is still needed to address these issues.

The second test case is a complicated 3D problem, i.e. the Boeing 747 case, with its surface grid illustrated in Fig. 5. The Mach number is 0.84 and the angle of attack is 2.7 degrees. From the partial close-up view in Figs. 6 and 7, it can be seen that the geometry of a Boeing 747 is complex and comprises a large number of unstructured cells that causes slow convergence of the solutions when using some linear equation solvers such as conjugate gradient, GMRES, Gauss-Seidel. Our parallel AMG

Table 1. Performance of multi-element airfoil

<table>
<thead>
<tr>
<th>No. of Processor</th>
<th>2</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>32</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed up</td>
<td>1.95</td>
<td>3.87</td>
<td>7.2</td>
<td>10.9</td>
<td>13.1</td>
</tr>
</tbody>
</table>

5 Conclusions

An implicit parallel AMG method has been developed, which is composed of two key components, i.e. a building strategy and an agglomeration strategy. The building strategy defines the methods for coarsening grid and building the hierarchy of grids. The agglomeration strategy
monitors the workload on processors and decides when to start subdomain merging. Then the subdomains are merged together and processed on a small number of processors at the coarsest level of grid to improve the overall performance. The numerical results based on our parallel strategies for 2-dimensional case agree well with the physical experimental results and also exhibits good parallel performance. A complex 3D flow past Boeing 747 was also employed to evaluate the proposed approach, which shows that our parallel AMG method achieved good parallel performance and the agglomeration strategy improved the speedup by 5% on average in this case.

Fig. 6 Close-up view of the forebody of Boeing 747

Fig. 7 Close-up view of the wing with two engines of Boeing 747

References:


Fig. 8 Speedup for Boeing 747 case