ASYNCHRONOUS PARALLEL PRECONDITIONERS FOR COMPUTATIONAL FLUID DYNAMICS

ADITYA KASHI*, SYAM VANGARA* AND SIVA NADARAJAH*
* Department of Mechanical Engineering
McGill University
Sherbrooke Street W., Montreal, QC H3A 0C3, Canada
e-mail: aditya.kashi@mail.mcgill.ca, syam.vangara@mail.mcgill.ca, siva.nadarajah@mcgill.ca

Key words: Implicit solvers, Preconditioning, Fine-grain parallelism, Asynchronous iterations

Abstract. We look at the application of asynchronous parallel iterations to the preconditioning of large sparse systems of equations arising in computational fluid dynamics in case of implicit temporal discretization. We propose block versions of the asynchronous preconditioners and investigate their effectiveness for some cases of external aerodynamics.

1 INTRODUCTION

The most computationally intensive task in computational fluid dynamics (CFD) solvers which use implicit temporal discretization is usually the solution of large systems of linear algebraic equations. With the introduction of specialized computing devices (such as general-purpose graphics processing units) dedicated to ‘massively parallel’ computing, it has been found that most established techniques for implicit solution of CFD equations cannot utilize this increased level of parallelism. It is important to take advantage of these highly parallel platforms because of their high throughput and energy efficiency in numerical computing.

Krylov subspace methods and multigrid schemes are important solvers in CFD. Many operations in Krylov solvers are inherently parallel, but a notable exception today is the preconditioner which tends to depend on traditional serial methods. Implicit smoothers for multigrid also face this issue. There is a need to investigate preconditioners and smoothers suited to both massively parallel hardware and CFD problems. We emphasize that we are interested in fine-grain parallelism within each node of a cluster, over and above parallelism across the nodes. In this work, we explore a class of techniques called asynchronous iterations. We illustrate their use in solving compressible flow problems of interest in aerodynamics, which in our knowledge, has not been investigated yet.

2 ASYNCHRONOUS ITERATIONS

In their paper, Chazan and Miranker [1] provide a theory for ‘chaotic relaxation’ of linear systems, known more generally as asynchronous iterations. This paper deals with asynchronous variants of stationary iterative methods. Their framework allows a situation
in which, for the update of any component, the latest available values of other components are used, which may or may not have been updated yet. This is useful when multiple processors are available, since there are no synchronization points. Chow and Patel propose [2] a nonlinear asynchronous iteration to compute ILU factorizations in parallel. We apply their asynchronous ILU(0) method to problems of external aerodynamics, and compare it to an asynchronous symmetric Gauss-Seidel (SGS) scheme. We also derive block variants of these schemes and show that they are more suitable (than the original asynchronous schemes) for compressible flows discretized with finite volumes.

As an example, we present results for a transonic turbulent (RANS) flow over a RAE-2822 airfoil (99280 cells on one 16-core Intel Sandy Bridge node). The wall-clock times and scaling are that of the total time taken by preconditioning operations over the entire nonlinear solve. The linear residual at each step is dropped by at most an order of magnitude, using at most 20 FGMRES iterations. Three asynchronous sweeps are used for each application of the block-SGS preconditioner.

![Figure 1: Execution speed as a percentage of the ideal, for block-SGS and (reference) block-Jacobi](image)

<table>
<thead>
<tr>
<th>Threads</th>
<th>Wall-clock time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (classic)</td>
<td>24.3</td>
</tr>
<tr>
<td>1 (async)</td>
<td>57.3</td>
</tr>
<tr>
<td>2 (async)</td>
<td>33.6</td>
</tr>
<tr>
<td>4 (async)</td>
<td>22.4</td>
</tr>
<tr>
<td>8 (async)</td>
<td>14.1</td>
</tr>
<tr>
<td>16 (async)</td>
<td>8.89</td>
</tr>
</tbody>
</table>

**Table 1:** Total preconditioner run times; all ‘async’ (asynchronous) runs use 3 sweeps for each preconditioner application

3 CONCLUSIONS

In the final presentation, we show that asynchronous parallel block SGS and ILU schemes provide significant speed-ups over their serial counterparts and discuss some notable aspects of these methods, including their use in multigrid solvers.

REFERENCES
