Performance Characterization and Scalability analysis of a Chimera Based Parallel Navier-Stokes solver on commodity clusters

A. Hamed\textsuperscript{a}, D. Basu\textsuperscript{a}, K. Tomko\textsuperscript{b}, Q. Liu\textsuperscript{b}

\textsuperscript{a}Department of Aerospace Engineering, University of Cincinnati, USA

E-mail\textsuperscript{a} : a.hamed@uc.edu, basud@email.uc.edu

\textsuperscript{b}Department of Electrical and Computer Engineering and Computer Science, University of Cincinnati, USA

E-mail\textsuperscript{b} : ktomko@ececs.uc.edu, liuqu@ececs.uc.edu

Keywords: Domain Connectivity, PEGASUS, Navier-Stokes, Implicit, Upwind Scheme, Cache Performance and Relative Speedup

1. Abstract

The present work focuses on the performance characterization and scalability analysis of a chimera based parallel Navier-Stokes (N-S) solver on a commodity cluster. The parallel code, FDL3DI, is MPI-based and the interconnectivity between the zones is obtained through the Chimera approach. In this strategy, the computational domain is decomposed into a number of overlapped sub-domains and each of the sub-domains is assigned to a separate processor. Each sub-domain is solved independently and communication between them is accomplished through the interpolation points in the overlapped region by explicit message passing using MPI libraries. The code’s performance is assessed on a commodity cluster of AMD Athlon processors in terms of relative speed up, execution time and cache analysis with different grids and domain sizes. Turbulent flow over a flat plate is considered as the benchmark problem for the current simulations. Three variants of the upwind biased Roe schemes, a second order central difference scheme as well as a high order compact differencing scheme with high order pade-type filtering are used for spatial discretization and they are compared with each other for relative execution time. Computed results indicate that the number
of subdomains as well as the type of spatial discretization scheme used significantly affects the execution time. The relation between cache performance and the number of zones as well as the numerical schemes is also explored in the present analysis.

2. Introduction and Methodology

Parallel computational fluid dynamics (CFD) has evolved significantly over the past few years\textsuperscript{1,2,3,4}. With the advent of high performance clusters, and the resulting increase in availability of computing cycles, large-scale parallel CFD computations are now realizable. Most of the commercial CFD codes are available in a parallel version. However, the performance of a parallel code is a combination of many parameters, including the code’s scalability across an increasing number of nodes\textsuperscript{5}. Also the number of zones, overlapped points and the numerical scheme influence the performance of the solver.

In the present analysis, an extensive performance characterization as well as scalability analysis of a Chimera based parallel Navier-Stokes solver is being carried out using commodity clusters. The main focus of the current work is to study the scalability of the solver, the cache analysis, its performance with varying number of sub-domains and also its performance with different numerical schemes. The parallel time accurate three-dimensional solver FDL3DI was originally developed at AFRL\textsuperscript{6}. In the Chimera based parallelization strategy\textsuperscript{4} used in the solver, the computational domain is decomposed into a number of overlapped sub-domains as shown in figure 1\textsuperscript{7}. An automated pre-processor PEGSUS\textsuperscript{8} is used to determine the domain connectivity and interpolation function between the decomposed zones. In the solution process, each sub-domain is assigned to a separate processor and communication between them is accomplished through the interpolation points in the overlapped region by explicit message passing using MPI libraries. The solver has been validated and proven to be efficient and reliable for a wide range of high speed and low speed; steady and unsteady problems\textsuperscript{3,9,10}. Basu et al.\textsuperscript{2} implemented different hybrid turbulence models in the solver and carried out extensive simulations of unsteady separated turbulent flows.

The Navier-Stokes equations written in strong conservation-law forms are numerically solved employing the implicit, approximate-factorization, Beam-Warming algorithm\textsuperscript{11} along with the diagonal form of Pullinam and Chaussee\textsuperscript{12}. Newton subiterations are used to improve temporal accuracy and stability properties of the algorithm. The solver has three variants of upwind biased Roe schemes and a second order central difference scheme for spatial discretization. Apart from that, high order compact differencing scheme and high order pade-type filtering schemes are also available in the code. All of the details of the different numerical schemes, as well as their relative merits and demerits are explained in references 3, 6-9, 10. The execution efficiency of the numerical schemes is assessed in the current study. In a prior investigation\textsuperscript{4}, the code’s performance was analyzed using two supercomputers, namely the IBM SP3 and the Silicon Graphics Origin 2000. However, in the current analysis, the performance study and the scalability analysis is being carried out in a commodity cluster with AMD Athlon 2000+ processors. The 2\textsuperscript{nd} order central difference scheme was used for spatial
discretization in the earlier analysis. In the current work, results are being obtained using different variants of the upwind biased Roe scheme (1st, 2nd and the 3rd order), 2nd order central scheme as well as a high order compact scheme in conjunction with high order pade type filtering. They are used for the spatial discretization of the governing equations. An extensive comparison is presented for the comparative study of the different schemes including an investigation of memory cache utilization. Detailed analysis of cache usage is carried out for the different schemes as well as different numbers and sizes of zones. Level 1 and Level 2 cache effectiveness is characterized using the PAPI performance monitoring API. A prior cache analysis of the 2nd order version of the solver is described in reference 14.

3. High order compact Scheme and Filtering

High order compact schemes (schemes with an accuracy of fourth order and higher) are used in DNS (Direct Numerical Simulations) and LES (Large Eddy Simulation) of turbulent flows. High order compact schemes are non-dissipative and due to their superior resolution power, they represent an attractive choice for reducing dispersion, anisotropy and dissipation errors associated with low-order spatial discretization. However, high order compact-difference schemes are susceptible to numerical instability non-diffusive nature and approximate high order filters are used in conjunction to overcome their susceptibility to unrestricted growth of spurious perturbations. The filter has been proven superior to the use of explicitly added artificial dissipation (as used in upwind biased and central schemes) for maintaining both stability and accuracy. Compact schemes do however incur a moderate increase in computational expense over the non-compact schemes. A schematic of the formulation aspects of the compact scheme and the filtering technique is shown in figure 2. In the current simulations, a fourth order compact scheme in conjunction with a sixth-order non-dispersive filter is used. The compact scheme used in the computations henceforth will be designated as C4F6.

4. Parallel Algorithm

A detailed description of the parallelization algorithm employed in the solver is given in references 15. The single Program Multiple Data (SPMD) parallel programming style is used for the parallelization strategy. The flow equations for each grid are solved independently in parallel and the interpolated boundary values are also updated in parallel. The boundary data is exchanged between processors and after that, on each processor, the Chimera boundary conditions and the physical boundary conditions are applied to the assigned grid. The code running on each processor is identical and the processor identification number is used to determine which grid is assigned to each processor. The number of processor utilized is equal to the number of zones/subdomains, which ensures that each zone has been assigned to an individual processor. The MPI message-passing library is used for inter-processor communication. Point-to-point communication using send and receive calls are used to exchange the Chimera boundary data between processors.
5. Results and Discussions

Computed results are presented for the relative execution times for the different schemes, the results for the relative speedup for the different domain sizes and the cache analysis for different schemes. Simulations are carried out for a flat plate geometry. The geometry and flow conditions correspond to the experimental upstream flow conditions of the backward facing step flow by Driver and Seegmiller\textsuperscript{16}. The Mach number for the present simulations is 0.128 and the Reynolds number is $0.83 \times 10^6/\text{ft}$. The computational domain consists of 200 grid points in the streamwise direction, 100 points in the wall normal direction and 20 grids in the spanwise direction. At the inflow plane, the pressure is extrapolated from the interior and the other flow variables were prescribed. The pressure at the outflow boundaries was set equal to the free stream value and the other variables were extrapolated from the interior through a first order extrapolation. The top of the computational domain is set as outflow boundary. In the span-wise direction, symmetric boundary conditions are applied.

For the present study, three cases are considered. The computational domain is divided into 8-zone, 16 zone and 20 zone domains respectively. Figure 3 shows the computational grid with the different zones and the overlapped regions. Figure 4 shows the computed boundary layer profile and its comparison with the available experimental data. It can be seen that there is an excellent agreement between the computed solution and the experimental data\textsuperscript{15}. The computations were carried out using 3\textsuperscript{rd} order Roe scheme. Figure 5 shows the comparison of the execution time (in seconds) with the number of processors for the different numerical schemes. The execution time is based on 1000 iterations for each scheme. The schemes include the 2\textsuperscript{nd} order central scheme, as well as 1\textsuperscript{st}, 2\textsuperscript{nd} and 3\textsuperscript{rd} order Roe schemes and the C4F6 scheme (4\textsuperscript{th} order compact scheme with 6\textsuperscript{th} order filtering). It can be seen from figure 3 that for all the schemes, there is a significant reduction in the execution time with the increase in the number of processors. However, among the four schemes, the 2\textsuperscript{nd} order central scheme takes the lowest execution times. This is because the computation in Roe schemes involves additional limiter calculations that are not present in the central scheme. Among the Roe schemes, the 3\textsuperscript{rd} order scheme takes the highest execution time, followed by the 2\textsuperscript{nd} and the 1\textsuperscript{st} order scheme. This is expected since the upwind biased Roe schemes have more computation compared to the central scheme and among the Roe schemes, the 3\textsuperscript{rd} order Roe scheme deals with a larger stencil in all the directions compared to the 1\textsuperscript{st} and 2\textsuperscript{nd} order Roe schemes. Comparing the C4F6 scheme with the other schemes it can be observed that the execution time is approximately 45% greater for the C4F6 schemes. This is due to the fact that the compact scheme in conjunction with filtering involves significantly more computation compared to the upwind type schemes. However, the compact scheme is commonly used with coarser grid discretization.

Figure 6 shows the comparison of the normalized execution time per grid point for all the different schemes. This is obtained by dividing the total execution time by the total number of grid points for the 8 zones, 16 zones and the 20 zones cases. The total number of grid points takes into account the effect of overlapping and also the interpolated boundary data points. It can be seen that for all the schemes, there is a significant reduction in the normalized execution time with the increase in the number of processors and there is also considerable difference in the normalized execution time between the different schemes. The trend is similar to that seen in figure 5.
Figure 7 shows the relative speedup of the computation with the number of processors. This is based on the 3rd order Roe scheme. The blue line gives the ideal/linear speedup. The green line shows best achievable speedup given the redundant calculation incurred due to grid overlap at Chimera boundaries, while the black line shows the present simulation case for the 3rd order Roe scheme. The results show both the redundant work as well as the inter-zone communication adversely affects the speed-up. Figure 8 through 10 illustrate the memory performance of different schemes with regard to the L1 data cache and L2 data cache. The data for the Roe schemes (1st order, 2nd order and 3rd order), Central scheme (2nd order) and the 4th order compact scheme are plotted by profiling the main subroutine for solving flow equations (the subroutine that deals with the bulk of the computation) in the solver. The compact scheme involves much more variables, which results in higher cache accesses. In figure 8, the miss rates of L1 & L2 cache are given on the top of corresponding bars. It shows that L1 data cache miss rate is lower than 4% for all the schemes, which means that the L1 cache can satisfy most of the data lookup. Figure 9 enlarges the cache performance for 1st order Roe scheme with 8 zones. A low L1 data cache miss rate of 3.6% is achieved and the L1 data cache misses are taken care of by the L2 cache, which has a hit rate of 66%. Combining with figure 8, it can be easily seen that all other schemes have similar performance. Figure 10 shows the cache behaviors with different sub grid sizes. Since a higher number of zones imply a smaller subgrid size, both the cache access number and the cache miss number decrease accordingly. However, the miss rates of two levels of caches do not show any significant variation.

6. Conclusions

Computed results are presented for the performance characterization and scalability analysis of a chimera based parallel N-S solver on a commodity cluster. Comparison of the performance of the schemes shows that the relative execution time and the normalized execution time per grid point is similar for the 2nd order central scheme and the three Roe schemes but significantly greater for the C4F6 scheme. The execution times for all schemes can be effectively improved by increasing the number of chimera zones and hence processors used for a simulation but at some loss of efficiency due to the redundant work imposed by the grid over-lap and inter-zone communication. The cache analysis shows that the compact scheme results in higher cache accesses compared to the Roe schemes and the central scheme but still achieves a low miss rate. All schemes are effectively utilizing the 1st level cache and have miss rates less than 4%. Cache behaviors with different sub grid sizes indicate that both the cache access number and the cache miss number decrease with increasing number of zones.

7. Acknowledgements

The computations were performed in the AMD Athlon Cluster at UC. The authors would like to acknowledge Mr. Robert Ogden at the Aerospace Engineering Department at UC for setting up the cluster and for installing the profiling software PAPI in the cluster. The authors would like to thank Dr. Don Rizzetta and Dr. Philip Morgan at AFRL for their help and useful discussions regarding the FDL3DI code.
8. References

Figure 1 Schematic of the domain connectivity for the parallel solver (7)

Figure 2 Schematic of compact scheme and filtering (7)

Figure 3 Computational grids for the flat plate with 8 zones

Figure 4 Comparison of computed boundary layer profile with experimental data

Figure 5 Effect of numerical scheme on the execution time and number of processors
Figure 6 Normalized execution time (per grid point) for different schemes

Figure 7 Comparison of relative speedup with increasing number of processors (3rd order Roe scheme)

Figure 8 Number of Cache miss and Cache access for different schemes (8-zones)

Figure 9 % Cache Miss for 1st order (8 zones)

Figure 10 Number of cache miss for different zones (3rd order Roe scheme)