Scalability of Semi-Implicit Time Integrators for Nonhydrostatic Galerkin-based Atmospheric Models on Large Scale Cluster

James F. Kelly and Francis X. Giraldo
Department of Applied Mathematics
Naval Postgraduate School
Monterey, CA, USA
Email: jfkelly@nps.edu and fxgirald@nps.edu

Gabriele Jost
Texas Advanced Computing Center
University of Texas at Austin
Austin, TX, USA
Email: gjost@tacc.utexas.edu

Abstract—In this paper we describe the current status of our ongoing effort to optimize the efficiency of a novel application package for Nonhydrostatic Unified Model of the Atmosphere (NUMA) when running on large cluster architectures. The linear solver within a distributed memory paradigm is critical for overall model efficiency. The goal of this work-in-progress is to investigate the scalability of the model for different solvers and determine a set of optimal solvers to be used for different situations. We will describe our novel approach and demonstrate its scalability on a variety of clusters of multicore node clusters. We also present performance statistics to explain the scalability behavior.

Keywords—atmospheric models, time integrators, MPI, scalability, performance;

I. INTRODUCTION

Current limited-area, or mesoscale, atmospheric models require modeling nonhydrostatic effects, while next-generation global models are moving toward the nonhydrostatic regime. The nonhydrostatic atmospheric models, which run at resolutions finer than 10 km, possess fast-moving acoustic and gravity waves. These fast moving waves require a stringent CFL condition if the equations are discretized explicitly. To mitigate this problem, semi-implicit (SI) time-integrators, based on a Schur complement technique, have been developed for our Nonhydrostatic Unified Model for the Atmosphere (NUMA). SI time-integrators, which discretize the linear, fast moving waves implicitly, while discretize the slower dynamical processes explicitly, require a linear solve at each time-step. These time-integrators, while only conditionally stable, require less computational effort than fully-implicit time integrators, which are unconditionally stable. However, engineering efficient semi-implicit time-integrators which scale to tens of thousands of cores remains a significant challenge.

The efficiency of this linear solve within a distributed memory paradigm is critical for overall model efficiency. Therefore, we analyze the performance of two different semi-implicit time-integrators within distributed memory architectures. The goal of our work-in-progress is to compare the scalability of these solvers, thus determining the optimal time-integrator as a function of both the test problem and the parallel architecture. The rest of the paper is structured as follows: In Section 2 we describe the NUMA application and our approach to parallelization. In Section 3 we discuss in more detail the semi-implicit time-integrators which we employ and how they effect the efficiency of NUMA model. In Section 4 we describe the systems and test cases used in our evaluation. Section 5 presents presents scaling results and corresponding performance statistics. We draw our conclusions in Section 6 and also describe our future plans.

II. THE NONHYDROSTATIC UNIFIED MODEL OF THE ATMOSPHERE (NUMA) MODEL

A. Governing Equations

NUMA utilizes the fully compressible, non-hydrostatic Euler equations in non-conservative form in a global setting. These equations have previously been considered within a semi-implicit framework for 2D flows [2] and more recently for 3D flows [5] in Cartesian geometries. In the present study, we consider three-dimensional flow (x-y-z) subject to gravitational and Coriolis forces

\[
\frac{\partial \rho'}{\partial t} + \mathbf{u} \cdot \nabla \rho' + \nabla P \rho_0 \cdot \mathbf{u} + (\rho' - \rho_0) \nabla \cdot \mathbf{u} = 0 \quad (1a)
\]

\[
\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} + \frac{1}{\rho' + \rho_0} \nabla P' + \frac{\rho'}{\rho' + \rho_0} \hat{\mathbf{r}} + \mathbf{f} \times \mathbf{u} = 0 \quad (1b)
\]

\[
\frac{\partial \theta'}{\partial t} + \mathbf{u} \cdot \nabla \theta' + \nabla \theta \cdot \mathbf{u} = 0 \quad (1c)
\]

where the unknown variables are \((\rho', \mathbf{u}', \theta')\), where \(\rho' = \rho - \rho_0\) is a density perturbation, \(\mathbf{u} = (u, v, w)\) is velocity, and \(\theta' = \theta - \theta_0\) is potential temperature perturbation. In Eq. (1), \(\nabla V\) denotes a vertical gradient and \(\hat{\mathbf{r}}\) is a unit radial vector. The reference states \(\rho_0\) and \(\theta_0\) are hydrostatic and time-independent. Defining a solution vector \(\mathbf{q} = (\rho', \mathbf{u}', \theta')\), Eq. (1) is written in condensed form as

\[
\frac{\partial \mathbf{q}}{\partial t} = S(\mathbf{q}) \quad (2)
\]
In this paper we describe the current status of our ongoing effort to optimize the efficiency of a novel application package for Nonhydrostatic Unified Model of the Atmosphere (NUMA) when running on large cluster architectures. The linear solver within a distributed memory paradigm is critical for overall model efficiency. The goal of this work-in-progress is to investigate the scalability of the model for different solvers and determine a set of optimal solvers to be used for different situations. We will describe our novel approach and demonstrate its scalability on a variety of clusters of multicore node clusters. We also present performance statistics to explain the scalability behavior.
where the source term $S(q)$ is a right-hand side which is discretized by the spectral element method.

B. Spectral Element Discretization

The spectral element method [1] decomposes the spatial domain $\Omega \subset \mathbb{R}^3$ into $N_e$ disjoint elements $\Omega_e$ via

$$\Omega = \bigcup_{e=1}^{N_e} \Omega_e$$

(3)

where we choose $\Omega_e$ as hexahedra, which provides 1) simple grid generation and 2) efficient (fast) evaluation of the necessary differentiation and integration operators. Within each element $\Omega_e$, a finite-dimensional approximation $q_N$ is formed by expanding $q(x, t)$ in basis functions $\psi_j(x)$

$$q_N(x, t) = \sum_{j=1}^{M_N} q_j(t) \psi_j(x)$$

(4)

where $M_N = (N + 1)^3$ is the number of nodes per element and $N$ is the order of the basis functions. The discrete solution $q_N$ is assumed continuous across inter-element boundaries. Basis functions are constructed as tensor products of Lagrange polynomials $\psi_i(x) = h_a(\xi) \otimes h_b(\eta) \otimes h_c(\zeta)$, where $h_a(\xi)$ is the Lagrange polynomial associated with the Legendre-Gauss-Lobatto (LGL) points $\xi_i$ and $(\xi, \eta, \zeta)$ are functions of the physical variable $x$. Approximating the prognostic vector $q(x, t)$ by a finite-dimensional approximation $q_N$ in Eq. (4), multiplying by a basis function $\psi_i$ and integrating over the domain $\Omega$ yields the weak form

$$\int_{\Omega} \psi_i \frac{\partial q_N}{\partial t} \, d\Omega = \int_{\Omega} \psi_i S(q_N) \, d\Omega$$

(5)

where we have replaced the index $i$ with $I$ to emphasize that Eq. (5) is now a global representation. Applying the Galerkin expansion given by Eq. (4) to Eq. (5) yields the matrix-vector equation

$$\frac{\partial q_I}{\partial t} = M_{IJ} S(q_I)$$

(6)

where the mass-matrix $M_{IJ} = \int_{\Omega} \psi_I \psi_J \, d\Omega$ is diagonal if the interpolation and integration points are co-located. This approximation is valid for $N \geq 4$ while incurring a small error in integration [4]. Denoting the right-hand size (RHS) of Eq. (6) by $R_I(q_I)$, Eq. (6) is expressed as

$$\frac{\partial q_I}{\partial t} = \bigwedge_{e=1}^{N_e} R_I^{(e)}(q_I)$$

(7)

where $\bigwedge_{e=1}^{N_e}$ denotes the global assembly, or direct stiffness summation (DSS) operator that maps local, element-wise coordinates $i$ to global coordinates $I$. Eq. (7) forms the core of the spectral element method, allowing local, element-wise information $q_i^{(e)}$ to propagate to adjacent elements via the DSS operator.

C. Parallelization

III. IMPACT OF TIME INTERATORS ON MODEL EFFICIENCY

The fast-moving acoustic and gravity waves in Eq. (1) make explicit time-integration unfeasible do to the stringent CFL restriction. Therefore, we have developed both 3D and 1D semi-implicit time-integrators [3] which allow much larger time-steps. Both time-integrators utilize a pseudo-Helmholtz decomposition, which reduces the poorly conditioned monolithic $5N_p \times 5N_p$ system matrix to a well-conditioned $N_p \times N_p$ matrix.

A. 3D Semi-Implicit Time-Integrator

In the 3D Semi-Implicit (3D-SI) time-integrator, both the horizontally and vertically propagating acoustic and gravity waves are discretized implicitly, resulting in following linear system which must be solved at each time-step:

$$\mathcal{H}_{3D} P_{H} = R_{H}$$

(8)

where $\mathcal{H}_{3D} = \mathcal{H}(D, D^T)$ is a linear, pseudo-Helmholtz operator consisting of gradient $D$ and divergence $D^T$ operators operating on the discretized pressure $P_{H}$ and $R_{H}$ is an effective source term. Eq. (8) is solved iteratively using a Krylov sub-space method, which requires global reductions.

B. 1D Semi-Implicit Time-Integrator

In contrast, the 1D Semi-Implicit (1D-SI) time-integrator, only the vertically propagating acoustic and gravity waves are discretized implicitly. Hence, the global $N_p \times N_p$ problem is reduced into $N_H$, independent linear systems of size $N_V \times N_V$ which are solved at each time-step

$$\mathcal{H}_{1D} P_{H}^{(i)} = R_{H}^{(i)} \leq i \leq N_H.$$  

(9)

where $N_H$ is the number of horizontal grid points, $N_V$ is the number of vertical grid points, and $N_p = N_H + N_V$. In Eq. (9), $\mathcal{H}_{1D} = \mathcal{H}(D_{1D}, D_{1D}^T)$ is a linear, pseudo-Helmholtz operator consisting of vertical gradient $D$ and divergence $D^T$ operators. When combined with a horizontal domain-decomposition, no inter-processor communication is required. In typical global NWP problems, $N_V \ll N_H$.

IV. SCALABILITY AND PERFORMANCE PERFORMANCE ANALYSIS

In this section we present timings and performance analysis statistics for the 1D and 3D integrators.

A. Test Systems

We conducted scalability experiments of the following systems:

- The Constellation Linux Cluster Ranger is located at the Texas Advanced Computing Center (TACC). The Ranger system consists of of 3,936 16-way SMP compute nodes. Each nodes has 4 2.3 GHz AMD Quadcore Opteron providing 15,744 AMD Opteron processors
A TAU performance analysis tool on the Lonestar system is required by the GMRES iterative solver. We employed the Kraken system for a run with 864 MPI processes. The timing statistics displayed in Figure 3 show that 77% of the time is spent in global communication caused by a huge number of calls to MPI_Allreduce short message size. The test case under consideration is therefore a good candidate for the usage of the 1D integrator, yielding better scalability as well as a shorter execution time than the 3D integrator.

V. CONCLUSIONS AND FUTURE WORK
From the experiments and that we have conducted so far we conclude that our implementation of the 3D integrator excessive MPI communication time due to global reductions, at least for certain sets of input data. In our future work we will focus on issues: We will investigate possible optimizations for the 3D integrator, including the usage of other Krylov subspace methods. Second, we will conduct many more experiments on different types of input data sets. Our goal is to derive characteristics of the application runtime behavior in order to choose the most efficient solver.

ACKNOWLEDGMENT
The experiments on the Kraken Cray XT5 system were conducted via the TeraGrid. We would like to thank the Texas Advanced Computing Center (TACC) of the University of Texas at Austin which supported us by providing compute time and valuable support for our experiments on the Ranger and the Lonestar Clusters.

REFERENCES
<table>
<thead>
<tr>
<th>%Time</th>
<th>Excl.</th>
<th>Incl.</th>
<th>#Call</th>
<th>Subr.</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>100.0</td>
<td>9</td>
<td>1</td>
<td>19</td>
<td></td>
<td>136913239 NUMA3D</td>
</tr>
<tr>
<td>93.9</td>
<td>32</td>
<td>1</td>
<td>4322</td>
<td></td>
<td>128629276 NUMA3D</td>
</tr>
<tr>
<td>93.9</td>
<td>32</td>
<td>1</td>
<td>4322</td>
<td></td>
<td>128629276 TIME_LOOP</td>
</tr>
<tr>
<td>93.5</td>
<td>54</td>
<td>1439</td>
<td>11516</td>
<td></td>
<td>88943 TIME_LOOP =&gt; TI_BDF2</td>
</tr>
<tr>
<td>93.5</td>
<td>54</td>
<td>1439</td>
<td>11516</td>
<td></td>
<td>88943 TI_BDF2</td>
</tr>
<tr>
<td>92.6</td>
<td>1,046</td>
<td>2,069</td>
<td>1439</td>
<td>1,97163E+06</td>
<td>88250 TI_BDF2 =&gt; SOLVE_GMRES_SCHUR</td>
</tr>
<tr>
<td>92.6</td>
<td>1,045</td>
<td>2,069</td>
<td>1439</td>
<td>1,97163E+06</td>
<td>88250 TI_BDF2</td>
</tr>
<tr>
<td>77.8</td>
<td>1,45438</td>
<td>1,45438</td>
<td>1,67138E+06</td>
<td>0</td>
<td>63 MPI_Allreduce()</td>
</tr>
<tr>
<td>75.4</td>
<td>2,010</td>
<td>1,43249</td>
<td>1,68276E+06</td>
<td>1,68276E+06</td>
<td>64 GLSC2_MPI</td>
</tr>
<tr>
<td>75.4</td>
<td>2,010</td>
<td>1,43249</td>
<td>1,68276E+06</td>
<td>1,68276E+06</td>
<td>64 SOLVE_GMRES_SCHUR =&gt; GLSC2_M</td>
</tr>
<tr>
<td>73.0</td>
<td>1,41238</td>
<td>1,41238</td>
<td>1,60276E+06</td>
<td>0</td>
<td>63 GLSC2_MPI =&gt; MPI_Allreduce()</td>
</tr>
</tbody>
</table>

**USER EVENTS Profile NODE 1, CONTEXT 0, THREAD 0**

<table>
<thead>
<tr>
<th>NumSamples</th>
<th>MaxValue</th>
<th>MinValue</th>
<th>MeanValue</th>
<th>Std. Dev.</th>
<th>Event Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.671E+06</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>0</td>
<td>Message size for all-reduce</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>Message size for broadcast</td>
</tr>
<tr>
<td>3</td>
<td>40</td>
<td>32</td>
<td>37.33</td>
<td>3.771</td>
<td>Message size for reduce</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>Message size for scatter</td>
</tr>
</tbody>
</table>

Figure 3. Timing Statistics for a run on 288 MPI Processes on Lonestar using the 3D Integrator


