Accelerated ANSYS Fluent: Algebraic Multigrid on a GPU

Robert Strzodka
NVAMG Project Lead
A Parallel Success Story in Five Steps
Step 1: Understand Application
ANSYS Fluent Computational Fluid Dynamics
Step 2: Identify Bottleneck in Coupled Solver of Incompressible NS

- Assemble Linear System of Equations
- Solve Linear System of Equations: $Ax = b$

Runtime:
- Accelerate this first (~ 33%)
- No (~ 67%)

Converged?
- Yes: Stop
Step 3: Parallelize Algorithm
Algebraic Multigrid (AMG)

\[ A^f x^f = b^f \]

\[ A^c e^c = r^c \]
Step 4: Create Library of Production Quality Parallel Iterative Solvers

- People (NVIDIA and ANSYS)
  - Assemble a great team
  - Collaborate closely

- Algorithms
  - Innovate with parallelism
  - Understand numerical tradeoffs

- Software
  - Invest in library design and testing
  - Optimize for GPUs
Step 5: Enjoy Acceleration
ANSYS Fluent 14.5 with nvAMG Solver

![Bar chart comparing solver times for Dual Socket CPU and Dual Socket CPU + Tesla C2075.](chart)

- **2 x Xeon X5650, Only 1 Core Used**
  - Dual Socket CPU: 2832 sec
  - Dual Socket CPU + Tesla C2075: 517 sec
  - 5.5x improvement

- **2 x Xeon X5650, All 12 Cores Used**
  - Dual Socket CPU: 933 sec
  - Dual Socket CPU + Tesla C2075: 517 sec
  - 1.8x improvement

**Helix Model**
- Helix geometry
- 1.2M Hex cells
- Unsteady, laminar
- Coupled PBNS, DP
- AMG F-cycle on CPU
- AMG V-cycle on GPU

**NOTE:**
- This is a performance preview
- GPU support is a beta feature
- All jobs solver time only
More about nvAMG
nvAMG Library - Interaction

- Supported matrix formats
  - Scalar and block CSR
  - Single and double precision

- Infrastructure
  - CUDA, Thrust
  - NVIDIA GPUs, tuned for Tesla K20X

- Integration
  - Dynamically linkable library
  - Public C interface with flexible text parameters
  - C++ plugin system for low-level extensions
nvAMG Library - Solvers

- Library of nested solvers for large sparse $Ax=b$

- Nesting creates a solver hierarchy, e.g.

- **Example solvers**
  - **Jacobi**, simple local (neighbor) operations, no/little setup
  - **BiCGStab**, local and global operations, no setup
  - **MC-DILU**, graph coloring and factorization at setup
  - **AMG**, multi-level scheme, on each level: graph coarsening and matrix-matrix products at setup
Solvers
Jacobi Solver – Trivial Parallelism

- Defect correction with preconditioner \( M \)

\[
x^{n+1} = x^n + M^{-1} (b - Ax^n)
\]

- In case of Jacobi

\[
M = \begin{bmatrix}
D_1 & & \\
& D_2 & \\
& & \ddots
\end{bmatrix}
\]

Ds may be small blocks themselves, e.g. 4x4
ILU Solvers – Coloring Enables Parallelism

- Incomplete LU factorization: $M = LU \approx A$
- Graph coloring allows **parallel setup and solve**
- With $m$ unknowns and $p$ colors, $m/p$ unknowns run in parallel
From Geometric to Algebraic Multigrid

\[
A_h x_h = b_h
\]

\[
R_{2h} \downarrow \quad P_{2h} \uparrow
A_{2h} x_{2h} = b_{2h}
\]

\[
R_{4h} \downarrow \quad P_{4h} \uparrow
A_{4h} x_{4h} = b_{4h}
\]
From Fine to Coarse Matrix

Aggregation

Sparse matrix-matrix product
Parallel Sparse Matrix-Matrix Product

- Galerkin product in AMG: $A_{2h} = R_{2h}A_hP_{2h}$
- In general: $A \times B = C$

Two parallel steps
- Find the number of non-zeroes per row of $C$
- Compute the columns indices and values per row of $C$
nvAMG results for different Ax=b
Hardware

- K20X
  - Kepler architecture, Tesla K20X GPU Accelerator

- C2090
  - Fermi architecture, Tesla C2090 GPU Accelerator

- 3930K(6)
  - Sandy Bridge architecture, Core i7-3930K, 6 cores
AMG Timings on Regular Discretizations

- CPU Fluent solver: AMG(F-cycle, agg8, DILU, 0pre, 3post)
- GPU nvAMG solver: AMG(V-cycle, agg8, MC-DILU, 0pre, 3post)

Lower is Better
**AMG Timings on Irregular Discretizations**

- **CPU Fluent solver**: AMG(F-cycle, agg8, DILU, 0pre, 3post)
- **GPU nvAMG solver**: AMG(V-cycle, agg2, MC-DILU, 0pre, 3post)

Lower is Better

- **Bar Chart**
  - X-axis: Airfoil (hex 784K), Aircraft (hex 1798K)
  - Y-axis: Values from 0 to 9
  - Varies by device (K20X, C2090, 3930K(6))
## ANSYS and NVIDIA Collaboration Roadmap

<table>
<thead>
<tr>
<th>Release</th>
<th>ANSYS Mechanical</th>
<th>ANSYS Fluent</th>
</tr>
</thead>
<tbody>
<tr>
<td>13.0 Dec 2010</td>
<td>SMP, Single GPU, Sparse and PCG/JCG Solvers</td>
<td></td>
</tr>
<tr>
<td>14.0 Dec 2011</td>
<td>+ Distributed ANSYS; + Multi-node Support</td>
<td>Radiation Heat Transfer (beta)</td>
</tr>
<tr>
<td>14.5 Oct 2012</td>
<td>+ Multi-GPU Support; + Hybrid PCG; + Kepler GPU Support</td>
<td>+ Radiation HT; + GPU AMG Solver (beta), Single GPU</td>
</tr>
<tr>
<td>15.0 Mid-2013</td>
<td>CUDA 5 + Kepler Tuning</td>
<td>Multi-GPU AMG Solver</td>
</tr>
</tbody>
</table>
A Parallel Success Story in Five Steps

- Step 1: Understand Application
- Step 2: Identify Bottlenecks
- Step 3: Parallelize Algorithms
- Step 4: Create Library
  - People (Team + Collaboration)
  - Algorithms (Innovation + Mathematics)
  - Software (Design + Optimization)
- Step 5: Enjoy Acceleration

Welcome to ANSYS Fluent with nvAMG
Starting with single GPU support as a beta feature in 14.5
Accelerated ANSYS Fluent: Algebraic Multigrid on a GPU

Robert Strzodka
NVAMG Project Lead