StreamFEM
A Streaming Language Implementation of the Discontinuous Galerkin Finite Element Method

Timothy Barth (NASA Ames)            Timothy Knight  (Stanford CS)

A DG Finite Element Method for Conservation Laws

StreamFEM implements the Discontinuous Galerkin (DG) finite element method for systems of nonlinear conservation laws in divergence form in 2-D or 3-D:

\[ u_t + \text{div}(f) = 0 \]

The DG Finite Element Variational Statement

Find \( u \in V_h \) such that \( \forall w \in V_h \)

\[
\sum_{e \in \text{elements}} \left( \int_K w u_t \, d\tilde{x} - \int_K \tilde{f}(u) \cdot \nabla w \, d\tilde{x} + \int_{\partial K} w \mathbf{n} \cdot (u_- - u_+) \, d\tilde{s} \right) + \int_K \epsilon_K(u) \nabla w \cdot \nabla u \, d\tilde{x} = 0
\]

with \( \epsilon_K(u) = h^{2-\beta} \| u_t + \text{div}(u) \|_{K} \), \( \beta \geq 0 \).

Variable Arithmetic Intensity

StreamFEM includes discontinuous Galerkin models of several representative nonlinear partial differential equation (PDE) systems of increasing complexity in 3-D:

- Scalar Advection (1 PDE)
- Euler Equations (5 PDEs)
- Magnetohydrodynamics (8 PDEs)

StreamFEM also includes various piecewise polynomial representations with an increasing number of degrees of freedom (dofs) ranging from piecewise constant to piecewise cubic polynomial approximation in 3-D:

- Piecewise constant elements (1 dof / (element-equation))
- Piecewise linear elements (4 dofs / (element-equation))
- Piecewise quadratic elements (10 dofs / (element-equation))
- Piecewise cubic elements (20 dofs / (element-equation))

By increasing the number of PDEs and the number of degrees of freedom per element, it is possible to alter the overall arithmetic intensity of the computation by 10x or more.

StreamFEM Flow Chart

StreamFEM has been implemented in the Brook stream language and later translated into StreamC/KernelC. The current algorithm utilizes a simple Runge-Kutta(1) time stepping algorithm:

For each timestep:

Loop over faces:
- Gather 2 element states
- Compute flux terms
- Store fluxes to memory

Loop over elements:
- Gather 4 flux terms
- Compute interior term and update element
- Store updated element state

Arithmetic Performance

<table>
<thead>
<tr>
<th></th>
<th>Constant (Euler)</th>
<th>Constant (MHD)</th>
<th>Linear (Euler)</th>
<th>Linear (MHD)</th>
<th>Quadratic (Euler)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GFLOPS</td>
<td>32.5</td>
<td>39.2</td>
<td>34.1</td>
<td>32.5</td>
<td>31.6</td>
</tr>
</tbody>
</table>

Arithmetic Intensity

<table>
<thead>
<tr>
<th></th>
<th>Constant (Euler)</th>
<th>Constant (MHD)</th>
<th>Linear (Euler)</th>
<th>Linear (MHD)</th>
<th>Quadratic (Euler)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ops per mem access</td>
<td>9.8</td>
<td>13.8</td>
<td>14.4</td>
<td>16.5</td>
<td>17.1</td>
</tr>
<tr>
<td>Ops per SRF access</td>
<td>3.7</td>
<td>5.1</td>
<td>4.6</td>
<td>5.5</td>
<td>5.0</td>
</tr>
</tbody>
</table>

Sustained Bandwidth

<table>
<thead>
<tr>
<th></th>
<th>Constant (Euler)</th>
<th>Constant (MHD)</th>
<th>Linear (Euler)</th>
<th>Linear (MHD)</th>
<th>Quadratic (Euler)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory BW</td>
<td>167.3</td>
<td>196.5</td>
<td>163.2</td>
<td>151.9</td>
<td>153.0</td>
</tr>
<tr>
<td>SRF BW</td>
<td>3.0</td>
<td>3.7</td>
<td>2.4</td>
<td>1.9</td>
<td>1.0</td>
</tr>
<tr>
<td>LRF BW</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>