A Classification of Scientific Visualization Algorithms for Massive Threading

Kenneth Moreland  Sandia National Laboratories
Berk Geveci  Kitware, Inc.
Kwan-Liu Ma  University of California at Davis
Robert Maynard  Kitware, Inc.

November 17, 2013
Cielo
AMD x86
Full x86 Core
+ Associated Cache
8 cores per die
MPI-Only feasible

Trinity (option 1)
NVIDIA GPU
2,880 cores collected in 15 SMX
Shared PC, Cache, Mem Fetches
Reduced control logic
MPI-Only not feasible
Extreme Scale is Threads, Threads, Threads!

<table>
<thead>
<tr>
<th></th>
<th>Jaguar – XT5</th>
<th>Titan – XK7</th>
<th>Exascale*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cores</td>
<td>224,256</td>
<td>299,008 and 18,688 gpu</td>
<td>1 billion</td>
</tr>
<tr>
<td>Concurrency</td>
<td>224,256 way</td>
<td>70 – 500 million way</td>
<td>10 – 100 billion way</td>
</tr>
<tr>
<td>Memory</td>
<td>300 Terabytes</td>
<td>700 Terabytes</td>
<td>128 Petabytes</td>
</tr>
</tbody>
</table>

- To succeed at extreme scale, you need to consider the finest possible level of concurrency
  - Expect each thread to process exactly one element
  - Disallow communication among threads

*Source: Scientific Discovery at the Exascale, Ahern, Shoshani, Ma, et al.*
Data Parallel Visualization

- Duplicate processes run independently on different partitions of data.
Data Parallel Visualization

- Duplicate processes run independently on different partitions of data.
Data Parallel Visualization

- Duplicate processes run independently on different partitions of data.
Data Parallel Visualization

- Duplicate processes run independently on different partitions of data.
Data Parallel Visualization

- Duplicate processes run independently on different partitions of data.
Key Principles

- Data Separability

- Mappable Input

- Result Invariant

[Law et al., 1999]
Mappable Input Can Break Down

Coarse Parallelism
Result reasonably divided among partitions.

Massive Parallelism
Most partitions are empty.
Result Invariant Can Break Down
Result Invariant Can Break Down

Repeated Vertices
Result Invariant Can Break Down
New Key Principles

Coarse Parallel

- Data Separability
- Mappable Input
- Result Invariant

Massive Parallel

- Data Separability
- Discoverable Input Mapping
- Collective Work
Characterization and Classification

Key Principles

- Data Separability
- Discoverable Input Mapping
- Collective Work

Characterization Parameters

- Separable Element
- Point Mapping
- Cell Mapping
- Field Mapping
- Collective Work
Basic Mapping

Input

Output
Basic Mapping

Input

Output
## Basic Mapping

<table>
<thead>
<tr>
<th>Separable Element</th>
<th>Any</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point Mapping</td>
<td>Identity</td>
</tr>
<tr>
<td>Cell Mapping</td>
<td>Identity</td>
</tr>
<tr>
<td>Field Mapping</td>
<td>1 to 1</td>
</tr>
<tr>
<td>Collective Work</td>
<td>None</td>
</tr>
</tbody>
</table>

**Diagrams:**
- Generate Ids
- Surface Vectors (with Glyph)
- Warp (of Slice)
Basic Mapping

functor()
Map by Cell
Map by Cell

Input $x_1, x_2, x_3, x_4$ 

Output $f(x_1, x_2, x_3, x_4)$
Map by Cell

Separable Element: Cell
Point Mapping: Identity
Cell Mapping: Identity
Field Mapping: Points on cell to cell
Collective Work: None
Map by Cell

functor()
Reconnect Cell
Reconnect Cell
Reconnect Cell

Input

Output
Reconnect Cell

Separable Element: Cell
Point Mapping: 1 to 0 or 1
Cell Mapping: 1 to 0 or more
Field Mapping: Identity
Collective Work: None (or find unused)
Reconnect Cell

Output

Keep these

Remove these
Build Independent Topology
Build Independent Topology
Build Independent Topology

**Separable Element**  Any

**Point Mapping**  1 element to many points

**Cell Mapping**  1 element to many cells (constant number)

**Field Mapping**  Identity

**Collective Work**  None
Build Connected Topology
Build Connected Topology
Build Connected Topology
Build Connected Topology

Separable Element  Cell
Point Mapping       1 cell to 0 or more points
Cell Mapping       1 to 0 or more
Field Mapping      Interpolated points
Collective Work    Resolve duplicate points

Contour          Slice          Tessellate
Build Connected Topology

Output

Merge points
Capture Cell Adjacencies
Capture Cell Adjacencies

\[ f(x_1, x_2, x_3, x_4) \]
Capture Cell Adjacencies

Separable Element: Point, edge, or face
Point Mapping: Identity
Cell Mapping: Identity
Field Mapping: Interpolated incident fields
Collective Work: Find incidence relationships

Extract Edges

Feature Edges
Gradient (with Glyph)
Normal Generation
Capture Cell Adjacencies

Incidence relationships may not be explicitly captured by topology structure.
Globally Reduce

\[ f(x_1, x_2, x_3, \ldots) \]
Globally Reduce

\[ f/(x_1, x_2, x_3, \ldots) \]
Globally Reduce

\[ f(x_1, x_2, x_3, \ldots) \]
## Globally Reduce

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Separable Element</td>
<td>Any</td>
</tr>
<tr>
<td>Point Mapping</td>
<td>None in output</td>
</tr>
<tr>
<td>Cell Mapping</td>
<td>None in output</td>
</tr>
<tr>
<td>Field Mapping</td>
<td>All to 1</td>
</tr>
<tr>
<td>Collective Work</td>
<td>Global Reduction</td>
</tr>
<tr>
<td>Algorithms</td>
<td>Histogram, Integrate, Outline (find bounds), Statistics</td>
</tr>
</tbody>
</table>
Query Data

Search Structure

Output

f(x)
Query Data

Separable Element: Point, key, or query
Point Mapping: Identity
Cell Mapping: Identity
Field Mapping: 1 query to 1 output
Collective Work: Building query structure

Stream Tracer
## Summary

<table>
<thead>
<tr>
<th>Name</th>
<th>Separable Element</th>
<th>Point Mapping</th>
<th>Cell Mapping</th>
<th>Field Mapping</th>
<th>Collective Work</th>
<th>Example Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Mapping</td>
<td>Any</td>
<td>Identity</td>
<td>Identity</td>
<td>1 to 1</td>
<td>None</td>
<td>Generate Ids</td>
</tr>
<tr>
<td>Map by Cell</td>
<td>Cell</td>
<td>Identity</td>
<td>Identity</td>
<td>Points on cell to cell</td>
<td>None</td>
<td>Cell Centers</td>
</tr>
<tr>
<td>Reconnect Cell</td>
<td>Cell</td>
<td>1 to 0 or 1</td>
<td>1 to 0 or more</td>
<td>Identity</td>
<td>None</td>
<td>Threshold</td>
</tr>
<tr>
<td>Build Independent Topology</td>
<td>Cell</td>
<td>1 element to many points</td>
<td>1 element to many cells</td>
<td>Identity</td>
<td>None</td>
<td>Glyph</td>
</tr>
<tr>
<td>Build Connected Topology</td>
<td>Cell</td>
<td>1 cell to 0 or more points</td>
<td>1 to 0 or more</td>
<td>Interpolated points</td>
<td>Resolve duplicate points</td>
<td>Contour</td>
</tr>
<tr>
<td>Capture Cell Adjacencies</td>
<td>Point, edge, or face</td>
<td>Identity</td>
<td>Identity</td>
<td>Interpolated incident fields</td>
<td>Find incidence relationships</td>
<td>Normal Generation</td>
</tr>
<tr>
<td>Globally Reduce</td>
<td>Any</td>
<td>None in output</td>
<td>None in output</td>
<td>All to 1</td>
<td>Global reduction</td>
<td>Histogram</td>
</tr>
<tr>
<td>Query Data</td>
<td>Point, key, or query</td>
<td>Identity</td>
<td>Identity</td>
<td>1 query to 1 output</td>
<td>Building query structure</td>
<td>Stream Tracer</td>
</tr>
</tbody>
</table>
Conclusion

- Accelerators and other emerging architectures use massive threading that takes parallelism to a whole different scale

- Visualization algorithms can be characterized by:
  - How data is separated
  - How input elements are mapped to output elements
  - What collective work is performed

- Algorithms that share all three characterizations can be implemented with similar mechanisms
  - We can exploit this observation to implement new algorithms